

Whole ALTO World Newsletter

Technology and Tools

XEROX

December 31, 1977

SPECIAL ANNOUNCEMENTS

WHOLE ALTO WORLD MEETING - The next Whole Alto World meeting is scheduled to be held from 9AM-3PM, February 7, 1977 at XEOS in Pasadena. Liz Bond is our hostess. The last page of the newsletter is a flyer announcing the meeting. Please detach your copy and post it on an appropriate bulletin board. See you all at the meeting!

GENERAL NOTES

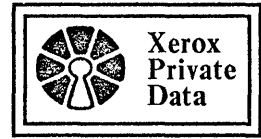
WRC JOINS THE NETWORK - On December 17th the line connecting the gateway at Webster with the gateway at PARC became operational. This, and the soon to be installed IFS, will promote the sharing of tools and information, much of which is currently under development at WRC. Attached to the newsletter is a diagram of the current network.

HARDWARE CATALOG - The first edition of the Alto Hardware Catalog is included with this issue of the newsletter. It lists by hardware type, e.g. Printers, Scanners, etc., devices that have been hooked to an Alto or the Ethernet. For an item to be included at least one working, reproducible copy must exist. Gathering this type of information tends to be a hit-or-miss proposition so if you know of an item that should be included, please contact the coordinator. It could save someone a lot of time and effort searching for or designing and building something that already exists.

TIME STANDARD CHANGE - The Alto's internal date and time standard is going to be changed because it is: 1) location dependent (WRC's Altos indicate PST), and 2) not monotonic (daylight saving time). The change will involve modifications to the Operating System and subsystems that deal with dates and times.

It is suggested that users update their disks promptly as subsystems are rereleased. Version 14 of the Operating System and a new release of BRAVO are expected at the end of January. Subsystems that incorporate the time standard changes, such as the afore mentioned BRAVO, will SWAT on the current Operating System (version 13). Old subsystems will continue to operate in the current manner on the new Operating System until April 30th when they begin reporting in GMT.

If you maintain such subsystems you should read <Taft>AltoTime.bravo and <Taft>Time.tty. Briefly, the current standard is a 32-bit integer denoting the number of seconds since midnight, January 1, 1901, PST. It is manually reset for Daylight Savings time as appropriate. The new standard will denote the number of seconds since midnight, January 1, 1901, GMT. This time will be modified by local time zone, begin daylight savings date, and end daylight savings date. These parameters will be kept in server hosts (gateways, IFSs, etc.) and in reserved locations in Alto main memory and on Alto disk (for the convenience of stand-alone Altos), and updated whenever a timeserver is accessible.



PROPRIETARY INFORMATION - The patent department, in response to specific questions of general interest and after consulting with appropriate members of management, has made the following recommendations on Alto, Dover, and Sequoia.

What can we tell job applicants during an interview?

Only information that Xerox has made public or is otherwise in the public domain.

Can we demonstrate Alto to outsiders using University of Rochester software?

Yes. (Ed note: A list of public software is being developed.)

Can Consultants and/or Co-op students work on Alto?

Yes, providing they each sign an appropriate agreement containing a confidential disclosure provision.

Can outsiders see Dover or Sequoia?

They may be shown the outside of the units providing each has signed an appropriate agreement containing a confidential disclosure provision. *The internals may not be shown.*

Can consultants be shown Dover or Sequoia?

Only information absolutely necessary to perform the consulting services may be shown, and then only if the consultant has signed an appropriate agreement containing a confidential disclosure provision.

Can the output from Dover or Sequoia be distributed outside Xerox?

Yes, provided the copy does not identify its source or method of generation.

TOOLS

HARDWARE

DEAD ALTO'S - The following text was received in a memo from Ron Cude this last month. It is filed on [MAXC]<SPG>DeadAlto.memo.

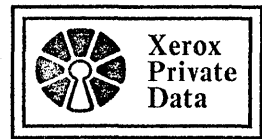
This memo is intended to point out a potential problem that some of you may be experiencing with your Alto II's. Have you ever had your Alto not want to re-boot after running out of the Cram? If so there may be a fix. To see if this is your problem, you will need two files:

Ramload.run and AltoIICode2.mb

Try the following:

Ramload AltoIICode2.mb/F 0/V cr

Ramload will display some information and ask for a confirm. After confirming, it will load the Cram with the .mb file. When it comes back and says Boot?, confirm with a carriage return. The .mb file is now running out of the Cram. Type Ramload/T and confirm with a cr. Ramload will write random numbers into the Cram blowing up your Alto (not literally, just the microcode). If you can now re-boot, you don't have the problem. If you can't re-boot, you have the problem and should investigate as shown below.



Check your Cram module (assy. #216365) to make sure that note K. (page 2 of the Cram module print) has been done. This note is an etch cut to hole next to label "R7". If this etch cut has not been made, it will blow one and possibly two components on the Disk Control module. If the cut has not been made, please make it and also install a new MPQ3303 in A1 on the Disk Controller and a new 74H00 in A11 also on the Disk Controller. If your Cram module already has the etch cut but you are experiencing the problem anyway, try replacing the same two components on the Disk Controller.

ALTO II ENGINEERING CHANGES - A list of the current revision levels for Alto II components was mailed this last month to the people that perform Alto maintenance. Future Engineering Orders will also be mailed as they are ready for distribution. If you maintain Altos and have not received the current revision level list, contact the coordinator.

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local IFS under the directories <Alto> and <AltoDocs>. If they are not available or if you are in doubt about the version, they may be retrieved from [MAXC] under the appropriate directory. Files stored under other directories are on [MAXC] unless otherwise indicated, such as [XEOS].

NEW RELEASE: CALCULATOR - This boot program from Joe Maleson pictures a TI SR-52 on the display which is operated by using the mouse to select the appropriate keys. Not all functions are implemented (i.e. it is not programmable). It may be run by booting the NetExec and entering "calculator" or by booting while depressing the <BS>, ";], and <blank-middle> keys. Documentation is the SR-52 manual.

NEW RELEASE: MICROFLOAT - Joe Maleson has made available a microcoded version of the FLOAT package. It is four to six times faster than the assembly language version (about 80 μ sec for multiply and divide, 40 μ sec for add and subtract). The documentation, <AltoDocs>Float.tty, is appended.

NEW RELEASE: SIGMA/ETH - This is a pair of subsystems by Thomas Holladay and Keith Knox to transfer arbitrary files between an Alto and a SIGMA 3 over the Ethernet. They will be made available on request. The documentation, a Xerox Internal Report "Ethernet Software for Data Transfer between the SIGMA 3 and an ALTO", Accession No. X7704459, has not been included with the newsletter.

NEW RELEASE: SPLINE - This is a set of packages by Patrick Baudelaire to both compute cubic splines and to map them onto an Alto display bitmap. The documentation, <AltoDocs>Spline.tty, is appended.

NEW RELEASE: TYPE - This is a small subsystem by Roger Bates for use in place of the Executive supplied "type.~" that displays a larger page, suppresses Bravo trailer information, can backup, etc. It can be retrieved from <Alto>Type.run and is documented under <AltoDocs>Type.tty. The documentation is appended.

ReReleases - Subsystems

COPYDISK - This version fixes a bug which prevented copying the second disk of a two-disk system. *The change makes this new version incompatible with previous version.* The new version is available on boot servers or may be retrieved from <Alto>CopyDisk.run(18-



Dec-77).

DMT - The new version has been enhanced for extended memory machines in addition to many internal changes. It is available from <Alto>DMT.boot(17-Dec-77) and boot servers (Gateways are boot servers). If you have access to a Gateway, simply delete DMT.boot from your disks and the executive will automatically retrieve it when you "quit".

EMPRESS - In addition to several bug fixes, the new version applies the /c option to press files, will merge two or more single page press files appending one additional press or text file if specified, and can personalize individual copies of a document. It is filed on <Alto>Empress.run(14-Dec-77). The documentation, <AltoDocs>Empress.tty (14-Dec-77), has been revised.

PEEKSUM - Peek disks should be rebuilt periodically using PeekDisk.cm. See the revised documentation <AltoDocs>DMT.tty.

TECHNOLOGY

Several methodologies of information retrieval are being investigated at the Webster Research Center. One of them, QUANSY, is an empirical natural language question-answering system by Avi Ben David. Currently operating at the fourth grade level over a narrow range of subjects (as measured by standard tests), the next level of QUANSY is expected to show substantially improved capability.

Three papers have been written and will soon be available as Xerox Internal Reports (Accession numbers are not yet assigned). The first, "A Parser Analyser of Empirical Design for Question-Answering", AFIPS Conference Proceedings, NCC, Vol. 46, pp. 669-678, is easily available and so is not reproduced here. The second paper, "A Memory Structure of Empirical Design for Question-Answering", presents the relationship between natural language and the empirically developed memory structures.

The most recent of the papers "Memory Interaction and Question-Answering in the QUANSY Question-Answering System" provides a sample dialog, an overview of the system, brief descriptions of the parser-analyser, memory structures and question-answering routines, and an in-depth discussion of concepts behind the interaction between the parser-analyser and memory structures.

Whole ALTO World Newsletter

Technology and Tools

XEROX

February 28, 1978

SPECIAL NOTE

NEW OPERATING SYSTEM - As previously announced, the timestandard implementation is being changed. This last week a new OPERATING SYSTEM and BRAVO were released. As future releases of this and other subsystems will not necessarily operate properly with the old operating system, *you should change over as soon as possible* by retrieving and executing NEWOS.cm from your local IFS or MAXC. You will need about 300 free pages on your disk. Check with your local support people for special procedures. The documentation, <AltoDocs>OS.press, has been revised.

GENERAL NOTES

WHOLE ALTO WORLD MEETING - The Whole Alto World meeting was hosted by Liz Bond of XEOS in Pasadena on February 7, 1978. Fifty-five people, representing virtually every Alto using group, attended.

The Distributed Message System(DMS), an upcoming, Alto-based replacement for the MAXC MSG system, was described by Frank Ludolph. Under DMS, messages are stored on IFS stations (or MAXC for the immediate future) only during transit. Received messages will be stored on the user's Alto disk in one or more user-designated files managed by Alto resident software. The user interface will be familiar to Alto users, consisting of several windows, menus, and Bravo style editing facilities. Although it is a research project, it is expected that DMS will be available to MAXC MSG users this summer.

Dick Sonderegger, SD Support, reports that MESA is now available through the Whole Alto World coordinator on a limited basis, by specific request, and depending on the proposed application. The language is still evolving and should not be used for long term development projects. Questions and problems should also be channeled through the coordinator's office to <SDsupport>.

Barry Smith, Sheldon Raizes, Terry Anderson, and Irv Keschner, lawyers with the Xerox Patent Department, attended the meeting to discuss the methods used to protect intellectual property, trade secrets, patents, and copyrights, as they apply to the Alto. Barry will be working with WAW in the near future to develop written material on this subject. The material will be printed in the Newsletter when it becomes available.

Terry Haney spoke on SPG's board repair activity. Boards should be sent to Terry, *along with a description of the problems* and, if from Orbit or Dover, a copy of the printer's output. Boards are logged and their repair scheduled in conjunction with SPG's other activities.

Jim Hall announced that his 1200 group is very interested in providing maintenance service for as many Altos as possible. Existing spares inventories can be turned in for credit. Contact Jim for pricing particulars.

The 7th Alto build will proceed on schedule according to Doug Stewart. This will be the last Alto build. There has been some difficulty obtaining 7000 bases for the Dover build (marketing has been quite successful in placing them recently), but it is not expected to significantly delay Dover deliveries.

Sam Losh of XEOS reports that Sequoia development is continuing. He requested that organizations interested in obtaining Sequoias contact him. If there is sufficient interest, deliveries could begin in the fall.

John Ellenby briefly described Advanced Systems Division's role in marketing test probes based on Alto technology. ASD has requested information on the Fuji Xerox mag brush developer, used on their 7200, for possible retrofit to Dover. The unit would improve solid area development. Additional information will be printed in the Newsletter as it is available.

The reasons for developing Altos as gateways were outlined by Ted Strollo. Essentially, the current Novas present maintenance problems, the small memory (32K) prevents further software development, and the Nova operating system is not as malleable as the Alto's. The number of gateways is expected to increase to as many as ten this year. Though the D0 will eventually be used in this capacity, they will not be available for this application for some time.

The meeting was then adjourned to permit attendees to see the Boca Raton Insurance tape, hosted by John Ellenby, and demonstrations of the touch screen (Dave Moulding), Smalltalk (Alan Kay), FIRST (Bob Datolla), and HSIL (Marion Suggs, Paul Lam).

ALTO MAINTAINERS MEETING - A meeting of Alto maintainers was held of February 8th, 1978 at El Segundo. The meeting was hosted by Doug Stewart, SPG. The primary subjects of discussion were hardware problem areas, centralized repair reporting, and SPG repair service.

The biggest problem area seemed to be the disk drives. Typical adjustments for the read gate are 460/440 n sec for the long/short one shots though this may vary from drive to drive. Also, the write head current is normally cut past track 128 due to the reduced track length. Cutting resistor F-63 on the J-10 board to raise write head current is common but Diablo advises against it suggesting instead that the value of resistor H-64 (part of the same voltage divider network) be varied starting with 1K and working down as necessary. The heads should be cleaned periodically (approximately 3 months) using a lint free material such as TexWipes. Q-Tips should not be used as they will leave fibers on the head. Alignment is generally performed after cleaning heads. Some groups keep spare heads for replacements.

The 15 volt Sorenson power supply (and to an extent the 12 volt supply) is the other major problem area. As these units are under a five year warranty they should be returned to the manufacture. Sorenson will also update units returned for repair.

It's useful to have a few memory chips on hand as this is the most common chip failure and it is easy to repair. Bad 16K memory chips should be returned to Terry Haney for failure evaluation. Memory Chips may be purchased from SPG. These are the only chips available from that group.

Keyboards have multiple character and mechanical sticking problems. The Keytest diagnostic can be used to adjust the key producing multiple characters. The electronics are on the AIM module and keyboard printed wiring board.

Modules will be repaired by SPG in conjunction with their other activities. No headcount is specifically assigned to repair activity. Boards should be returned to Terry Haney *along with a description of the problem* and, if applicable, a copy of printer output.

There is considerable interest in developing a repair data base. The only data of this type currently available is maintained by Jim Hall's 1200 group on the machines maintained by them under contract. Doug Stewart will set up a mechanism for collecting failure information including net address, date, subsystem affected, serial number (if applicable), failure symptoms, and corrective actions. Jim Iverson reports that a paper log is currently being kept for each machine in his group for the convenience in the multiple user environment and to identify recurring failures in a specific unit.

It was requested that Frank Ludolph set up a system to more quickly disseminate maintenance information.

ALTO MAINTAINER'S MESSAGE LIST - An immediate result of the Alto maintainers meeting is the establishment of a MAXC MSG distribution list file, <Secretary>AltoMaintainers.msg, to simplify the communication of general interest information among Alto maintainers. To use this feature when sending a message, the response to "TO:" is "**↑b** <Secretary>AltoMaintainers.msg CR CR". The rest of the sndmsg procedure is normal. The message will be sent to all accounts listed in the .msg file.

MESSAGING FOR SPECIAL INTEREST GROUPS - The same messaging mechanism referred to in the preceding item is used by several special interest groups including; AIS, PROM (ProLog users), SIL, and AltoMaintainers. If you are actively involved in any of these are sndmsg to Jennette <Jenkins> for inclusion. A complete listing of all distribution lists can be retrieved from [MAXC]<Secretary>All.masterlist.

DON'T LEAVE YOUR DISK IN AN ALTO - As pointed out in a recent issue of SDD's Random Items, a disk is locked inside the Alto's disk drive when power is removed from the unit or when the 15 volt power supply fails. Failure of this supply is one of the most common Alto ailments. *It is suggested that you not leave your disk in an Alto overnight.*

TOOLS

HARDWARE

NEW HARDWARE MANUAL - The Alto Hardware manual has been revised and made available in Press format. If your print server does not have two disk drives, the file <AltoDocs>AltoHardware.press may have to be broken into two pieces using PRESSEEDIT and the pieces sent to the printer.

ORBIT BUG - Severo Ornstein reports that there is a timing problem in the Orbit adapter. It appeared that Pimlico had problems aligning the successive color passes on a page. In reality the Output Scanline counter (SLN/SLWN) wasn't resetting properly due to a race situation resulting from anding the clock pulse with the clear level using an N163 (synchronous clear). This situation also exists with Dover but isn't very noticable because it shifts the image by only a fraction of a band.

The fix is simple; replace the three N163s in locations G2, G3, and G4 on the Input board with N161s. *Severo suggests that the fix be made on all Orbits*, regardless of attached printer, because it could create a very difficult bug to locate someday if printers are exchanged.

Whole ALTO World Newsletter

4

MAKING A DUAL-DRIVE ALTO - Doug Stewart has written a memo listing the items necessary to connect a second drive to the Alto. All items can be ordered directly from Diablo. The memo is appended to the Newsletter.

DISK DIAGNOSTIC DOCUMENTATION - Jim Cucinitti recently wrote some documentation for the Model 31 disk diagnostics that have been in use for quite sometime. It describes diagnostic initiation, use of the debugger, understanding the failure data, and modification of the diagnostics. *It is intended for maintainers only.* The document, which includes assembler listings of the programs, can be retrieved from [MAXC]<AltoDocs>DiskDiag.press.

PROM DESTRUCTION BY THE PROLOG PROGRAMMER - Tom Chang informs us that the ProLog PROM programmer will often destroy the last location of a PROM in socket PM when powered down. The ProLog people advise that *PROMs should always be inserted and removed from the socket with the power on.*

COLOR DISPLAYS AND THE ALTO - Every now and then Dick Shoup is asked about the use of color displays with the Alto. While it has been done, the results were generally unsatisfactory. Dick has written a memo on this topic which has been attached to this Newsletter.

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local IVY server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [XEOS].

NEW RELEASE: CONDENSE.RUN - This recently released program by Keith Knox will retrieve the screen bitmap from the SWAT and SWATEE files. The bitmap can be displayed or output to a file in either AIS or PRESS format. Documentation will be forthcoming shortly, but isn't really required for operation as the menus tell all. Retrieve [WRC]<IPA>Subsystems>Condense.run.

NEW RELEASE: AISdump.run - A new dump program, part of the AIS System, will write out the pixels as decimal values for an 8 bit/pixel or 1 bit/pixel AIS file to a file on the Diablo disk. Since the dump file is a text file, it is a 4:1 or 16:1 expansion, so be careful how large a window you choose. Retrieve [WRC]<AIS>Subsystems>AISdump.run.

ReReleases - Subsystems

AISmagnify - This new version, 2.0, has a menu, runs a little faster, and some new features. Retrieve [WRC]<AIS>Subsystems>AISmagnify.run. The documentation is [WRC]<AIS>MEMOS>AISmagnify.press.

BRAVO - The new version, 7.1, contains bug fixes and implements the new time standard. It will be retrieved and installed automatically when installing the new operating system. Documentation on the new color facilities can be retrieved from [IRIS]<Bravo>ColorBravoChanges.bravo.

CHAT - This rerelease, TTY version 9, Display version 15, contains bug fixes and minor improvements. Retrieve <Alto>Chat.run. The documentation, Chat.tty, is updated to include the I and O commands which toggles the USER.cm entry TYPESCRIPTCHARS.

COPYDISK - This subsystem, found on boot servers, has been updated to include the new time standard.

DMT - This subsystem, found on boot servers, has been updated to include the new timestandard. Also, the bug in the Dec. 10 version, which fails to indicated the bad RAM chip location, has been corrected.

IFS - The new release, 1.14, includes commands for accessing and manipulating file protections. Users should retrieve and read <IFS>HowToUse.press.

PRESSEEDIT - An experimental release of this subsystem can be found on <Newman>PressEdit.run. The documentation PressEdit.tty is on the same directory. It provides a new, simpler method of combining illustrations with text documents. Official release will occur in March after sufficient testing. The experimental version is reasonable robust.

PROM - The nature of the changes is unknown to me. Retrieve <Alto>PROM.run. New documentation is available on <EOD>PROM.bravo.

SCAVENGER - The nature of the changes is unknown to me. The documentation is unchanged. Retrieve <Alto>Scavenger.run.

SETTIME - The new version implements the new timestandard. Is is automatically retrieved when installing the new operating system with NewOS.cm.

SIL - Several changes have been made and are summarized in <SIL>SILupdates.press. Retrieve <SIL>SIL.run. The documentation SILmanual.press and SILsummary.press, also on <SIL>, have been revised.

ReReleases - Packages

ALTODEFS, ALTOFILESYS, DISKS, STREAMS, SYSDEFS - These definition files have changed in conjunction with the new operating system. If they currently reside on a disk they will be updated when NewOS.cm is run. For a description of changes see the change history in the rereleased OS Manual.

TECHNOLOGY

This month's paper, *GUS, A Frame-Driven Dialog System* by Daniel Bobrow, Ronald Kaplan, Martin Kay, Donald Norman, Henry Thompson, and Terry Winograd, is the third in a series on methods of making machines more amenable to the naive user. While this work is strictly research and not being performed on Altos, it gives us a glimpse of possible future directions.

The Understander project at PARC is exploring the process of language comprehension and the cognitive structures and operations which underlie it. GUS was written to assess progress and suggest area of future effort.

GUS, A Frame-Driven Dialog System

Daniel G. Bobrow, Ronald M. Kaplan, Martin Kay,
Donald A. Norman, Henry Thompson, Terry Winograd¹

*Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, California 94304*

GUS is the first of a series of experimental computer systems that we intend to construct as part of a program of research on language understanding. In large measure, these systems will fill the role of periodic progress reports, summarizing what we have learned, assessing the mutual coherence of the various lines of investigation we have been following, and suggesting where more emphasis is needed in future work. GUS (*Genial Understander System*) is intended to engage a sympathetic and highly cooperative human in an English dialog, directed towards a specific goal within a very restricted domain of discourse. As a starting point, GUS was restricted to the role of a travel agent in a conversation with a client who wants to make a simple return trip to a single city in California.

There is good reason for restricting the domain of discourse for a computer system which is to engage in an English dialog. Specializing the subject matter that the system can talk about permits it to achieve some measure of realism without encompassing all the possibilities of human knowledge or of the English language. It also provides the user with specific motivation for participating in the conversation, thus narrowing the range of expectations that GUS must have about the user's purposes. A system restricted in this way will be more able to guide the conversation within the boundaries of its competence.

MOTIVATION AND DESIGN ISSUES

Within its limitations, GUS is able to conduct a more-or-less realistic dialog. But the outward behavior of this first system is not what makes it interesting or significant. There are, after all, much more convenient ways to plan a trip and, unlike some other artificial intelligence programs, GUS does not offer services or furnish information that are otherwise difficult or impossible to obtain. The system is interesting because of the phenomena of natural dialog that it attempts to model and because of the principles of program organization around which it was designed. Among the hallmarks of natural dialogs are unexpected and seemingly unpredictable sequences of events. We describe some of the forms that these can take below. We then go on to discuss the modular design which makes the system relatively insensitive to the vagaries of ordinary conversation.

¹ This work was done by the language understander project at the Xerox Palo Alto Research Center. Additional affiliations: D. A. Norman, University of California, San Diego; H. Thompson, University of California, Berkeley; and T. Winograd, Stanford University. To appear in *Artificial Intelligence*, Spring 1977 (8:1)

Problems of natural dialog

The simple dialog shown in Figure 1 illustrates some of the language-understanding problems we attacked. (The bracketed numbers are for reference in the text). The problems illustrated in this figure, and described in the paragraphs below, include: allowing both the client and the system to take the initiative, understanding indirect answers to questions, resolving anaphora, understanding fragments of sentences offered as answers to questions, and interpreting the discourse in the light of known conversational patterns.

Mixed Initiative. A typical contribution to a dialog, in addition to its more obvious functions, conveys an expectation about how the other participant will respond. This is clearest in the case of a question, but it is true of all dialog. If one of the participants has very particular expectations and states them strongly whenever he speaks, and if the other always responds in such a way as to meet the expectations conveyed, then the initiative remains with the first participant throughout. The success of interactive computer systems can often be traced to the skill with which their designers were able to assure them such a dominating position in the interaction. In natural conversations between humans, however, each participant usually assumes the initiative from time to time. Either clear expectations are not stated or simply not honored.

GUS attempts to retain the initiative, but not to the extent of jeopardizing the natural flow of the conversation. To this extent it is a mixed-initiative system (see Carbonell, 1970a, 1970b). This is exemplified in the dialogue at [1] where the client volunteers more information than GUS requested. In addition to his destination, the client gives the date on which he wants to travel. Line [3] illustrates a case where the client takes control of the conversation. GUS had found a potentially acceptable flight and asked for the client's approval. Instead of either giving or denying it, the client replied with a question of his own.

Indirect answers. It is by no means always clear what constitutes an answer to a question. Frequently the purported answer is at best only a basis on which to infer the information requested. For example, when GUS asks "What time do you want to leave?" it is seeking information to constrain the selection of a flight. The client's response to this question, at [2], does constrain the flight selection, but only indirectly. In general, the kinds of inference necessary to interpret indirect answers can be very complex. GUS can handle the problem in its simpler manifestations, thanks largely to the very narrow expectations it has about the subject matter and the client's goals.

Resolving anaphora. A central issue in dialog is ensuring that both parties understand a given word or phrase as referring to the same object or event (see Norman and Rumelhart, Chapter 3, 1975 for a discussion of the general problem of reference). One problem occurs when a short phrase refers back to something that was introduced earlier. GUS provides for this problem in some of its simpler manifestations. At line [3], for example, the client's query refers to the flight mentioned in GUS's immediately preceding utterance. In [4] there is a reference to the flight mentioned earlier in the conversation, at [2]. Note that "next flight" in [3] was to be interpreted relative to the order of flights in the airline guide whereas "first one" in [4] refers to the order in which the flights were mentioned. Another

implicit referent underlies the use of "Friday" to specify a date in [6]. Resolution of this reference requires some complicated reasoning involving both the content and the context of the conversation. Since May 28th has been given as the departure date, it must presumably be the following Friday that the client has in mind. On the other hand, suppose that the specifications were reversed and Friday had been given as the departure date at line [1]. It would then be most readily interpretable as referring to the Friday immediately following the conversation.

GUS: Hello. My name is GUS. I can help you plan a simple trip by air.
Where do you want to go?

[1] *Client*: I want to go to San Diego on May 28

GUS: What time do you want to leave?

[2] *Client*: I must be in San Diego before 10 AM

GUS: Would you like P.S.A. flight 102 that arrives at 9:15 AM?

[3] *Client*: What is the next flight?

GUS: Air California flight 310 that leaves at 8:30 AM and arrives at 10:00 AM

[4] *Client*: I'll take the first one

GUS: In what name should I make a reservation, please?

[5] *Client*: Dan Bobrow

GUS: I have confirmed the following flight:
P.S.A. flight 102 on Wednesday May 28 from San Jose to San Diego
that leaves at 7:30 AM and arrives at 9:15 AM.
What date do you want to return on?

[6] *Client*: On Friday in the evening

GUS: Would you like the flight that leaves at 7:45 PM?

[7] *Client*: That's fine.

GUS: I have confirmed the following flight:
P.S.A. flight 307 on Friday May 30 from San Diego to San Jose
that leaves at 7:45 PM and arrives at 9:30 PM
Thank you for calling. Goodbye.

Figure 1. A transcript of an actual dialog with GUS

Sentence fragments. Utterances in natural conversation are by no means always complete sentences. This is not to say that there are no rules governing the ways in which fragments can be used. We collected a number of dialogs between people and examined the sentence

fragments that occurred: most appeared as answers to direct questions. Furthermore, a rule can almost invariably be derived from a question that will convert a fragmentary answer into a complete sentence expressing the same information. For example, the client's response in [5] to the request for a name is not a sentence but, when inserted in the blank space in the skeleton "You should make the reservation in the name of _____", it yields a sentence. Normal processing of the sentence so constructed gives the required interpretation of the fragment. This works even for the fragment in [6] which is not even a complete phrase.¹

¹The SRI speech system (Walker, et al., 1975) uses a number of other techniques for handling a different set of fragments.

These skeletons are systematically related, in the sense of transformational grammar, to the corresponding questions. The blank space in the skeletons usually occurs at the end. If Sgall and the linguists of the modern Prague school are right, then this follows from a strong tendency to organize sentences so that given information comes at the beginning and new information at the end. In this case, the given information is clearly that which is shared by the question and its answer.

Conversational patterns. Conversations conform to patterns, which are still only poorly understood, and there are specialized patterns that are used in special circumstances such as those that obtain in a travel agency. Realism requires that GUS fit its conversational strategy to these patterns. For example, flights are usually specified by departure time, but in response to [2], GUS specifies an arrival time, because the client had specified the arrival time to constrain the choice of flights. This is in accordance with a typical conversational convention; a speaker says as little as will suffice to communicate the point to be made. Grice [1975] calls these conventions conversational postulates and implicatures.

It seems also to be important to use conversational implicatures with respect to the goals of the client and the system in interpreting and generating the dialog (see Gordon & Lakoff (1972) for a general discussion of this issue). For example, in [1] the client says where he wants to go. GUS interprets this as a request for an action, that is, inserting the appropriate information into the travel plan being generated.

Principles of program organization

One of the major methodological issues we addressed in designing and building GUS was the question of *modularity*. We realize that language understanding systems, and other systems exhibiting some degree of intelligence, will be very large and complicated programs, and the flow of processing within them will be correspondingly complex. As Simon (1969) has pointed out, one way of reducing the complexity of a system is to decompose it into simpler, more readily comprehensible parts, and to develop and debug these in isolation from one another. When the separate modules have been constructed, however, the task of integrating them into a single system still remains. This can be difficult: truly complex systems are more than just the sum of their parts. The components, when put together, interact in subtle

but important ways. We implemented GUS in order to determine whether a modular approach for a dialog system was at all feasible and to test our notions of what reasonable lines of decomposition might be. We are aware of alternative decompositions, and are not committed to this one; it was convenient given the program modules already available, and the issues we wished to focus on. GUS provided a context in which to explore tools and techniques for building and integrating independent modules.

The major knowledge-oriented processes and structures in GUS--the morphological analyzer, the syntactic analyzer, the frame reasoner, and the language generator--were built as independent processes with well defined languages or data structures to communicate across the interfaces. They were debugged separately, and tied together by means of an overall asynchronous control mechanism.

Control: The organization of the system is based on the view that language-understanding systems must operate in a multiprocess environment (Kaplan, 1973b, 1975). In a system with many knowledge sources and a number of independent processes, some part of the mechanism must usually be devoted simply to deciding what shall be done next. GUS puts potential processes on a central *agenda*. GUS operates in a cycle in which it examines this agenda, chooses the next job to be done, and does it. In general, the execution of the selected task causes entries for new tasks to be created and placed on the agenda. Output text generation can be prompted by reasoning processes at any time, and inputs from the client are handled whenever they come in. There are places at which information from a later stage (such as one involving semantics) are fed back to an earlier stage (such as the parser). A supervisory process can reorder the agenda at any time. This process is similar in function to the control module in the BBN Speechlis system (Woods, 1974; Rovner, Nash-Webber & Woods, 1974), except that it can resume processes which are suspended with an active process state. Preserving the process state is necessary because the flow in the system is not unidirectional: for example, the state of the syntactic analysis cannot be completely abandoned when domain dependent translation starts. If a semantically and pragmatically appropriate interpretation of an utterance cannot be found from the first parsing, the syntactic analyzer must resume where it was suspended. INTERLISP's coroutine facility makes it possible to completely preserve the active state of the various processes (Teitelman, 1976; Bobrow & Wegbreit, 1974).

Procedural attachment. Broadly speaking, procedural attachment involves redrawing the traditional boundary between program and data in such a way as to give unusual primacy to data structures. Most of the procedures that make up a program, instead of operating on separate data structures, are linked to those structures and are activated when particular items of data are manipulated in particular ways. This technique lies at the heart of the reasoning component which is described in more detail later. It provides a natural way of associating operations with the classes or instances of data on which they are to operate. It is in some ways extensions of ideas found in SIMULA (Dahl & Nygaard, 1966) and SMALLTALK (Goldberg and Kay, 1976).

Monitoring and debugging: In a multiprocessing system with processes triggered by procedures attached to complex data structures, special tools are needed for programmers to

monitor the flow of control and changes in the data structures. Tightly linked with the agenda scheduler there is a central monitor with knowledge about how to summarize the current actions of the system. The monitor interprets special printing instructions associated with potential actions and particular items of data. In effect, the principle of procedural attachment has been extended to debugging information.

External data-bases: We believe that an important application of specialized dialog systems like GUS may be to help users deal with large files of formatted data. In the travel domain, the *Official Airline Guide* is such an external data-base. GUS can use an extract of this data-base, but the information in the file does not form part of its active working memory for the same reason that the information in the *Official Airline Guide* does not have to be memorized by a travel agent. Only that portion of the data base relevant to a particular conversation need be brought into the working memory of the system.

PROCESSES AND KNOWLEDGE BASES

Figure 2 illustrates the knowledge structures and processes in GUS. Each numbered row corresponds to a single knowledge based process in the system. The input to each process is shown in the left hand column. Each input is labelled with a number in parentheses indicating the row number of the process which produces it. Processes usually provide input to the ones listed below them. The third column names the process which produces the output structures specified in the fourth column, using for the processing the permanent knowledge bases specified in column two.

Input Structures	Permanent Knowledge Structures	Processes	Output Structures
1. Text String word (input) structures	Stem dictionary; Morphological rules	Dictionary lookup; Morphological analysis	Chart of data
2. Query context(6); of a Chart(1) sentence	Transition net grammar	Syntactic analysis	Parsing
3. Parsing of a frame sentence (2)	Case-frame dictionary	Case-frame analysis	Case- structure
4. Case-frame Frame change structure (3) description	Speech patterns; Domain specific frame forms	Domain dependent translation	
5. Frame change Frame change descriptions(4,5); descriptions Current frame response instances (5) descriptions;	Prototype frames and attached procedures	Frame reasoning	Output Current

frame

instances

6. Output response	Dialog query map;	Response	
English text;			
description (5)	Flight description	generation	Query
context			
	template		

Figure 2. Knowledge structures and Processes in GUS

Figure 3 shows the output structures of the earlier stages of processing of the sentence "I want to go to San Diego on May 28". Starting with an input string of characters typed by the client, a sequence of words is identified by a lexical analyzer consisting of a dictionary lookup process and a morphological analysis. The analysis program has access to a main dictionary of more than 3,000 stems and simple idioms and a body of morphological rules specifying how the information in the dictionary can be used to partition character sequences into known lexical items (Kay & Kaplan, 1976). The output of this stage is a *chart* (Kay, 1973), a table of syntactic and semantic information for use by the parser.

CLIENT: I want to go to San Diego on May 28

```
[S MOOD =DCL                                ...the syntactic analysis of the input
  SUBJ =[NP HEAD =[PRO CASE =NOMIN NUMBER =SG ROOT =I]]
  FVERB =[V TENSE =PRESENT ROOT =WANT] HEAD =WANT
  OBJ =[S MOOD =FOR-TO
        SUBJ =I
        HEAD =[V TENSE =PRESENT ROOT =GO]
        MODS =(
          [PP PREP =[PREP ROOT =TO]
            POBJ =[NP HEAD =[NPR PROPERTYPE =CITY-NAME
                          ROOT = SAN-DIEGO]]]
          [PP PREP =[PREP ROOT =ON]
            POBJ =[NP HEAD =[NPR PROPERTYPE =DATE-NAME
                          MONTH =MAY DAY =28]]]]]]]
```

```
[CLIENT DECLARE                                ...the case-frame structure
  (CASE FOR WANT/E (TENSE PRESENT)
    (AGENT (PATH DIALOG CLIENT PERSON))
    (EVENT (CASE FOR GO (TENSE PRESENT)
      (AGENT (PATH DIALOG CLIENT PERSON))
      (TO-PLACE (CASE FOR CITY
        (NAME SAN-DIEGO)))
      (DATE (CASE FOR DATE
        (MONTH MAY))
```

(DAY 28]

```

CMD: [ CLIENTDECLARE           ... the domain dependent translation, a
      (FRAME ISA TRIP-LEG      ... frame change description
      (TRAVELLER (PATH DIALOG CLIENT PERSON))
      (TO-PLACE (FRAME ISA CITY
                (NAME SAN-DIEGO)))
      (TRAVEL-DATE (FRAME ISA DATE
                  (MONTH MAY)
                  (DAY 28])

```

Figure 3. Processing the client's first utterance

The syntactic analyzer is based on the General Syntactic Processor (Kaplan, 1973a). Using a transition-network grammar and the chart, the parser builds one or more canonical syntactic structures, depending on whether or not the sentence is syntactically ambiguous. It finds one parse, and can continue to find others if the sentence is ambiguous and the first parse is rejected as uninterpretable by a later process. The syntactic analysis of the input sentence is shown in Figure 3.

The case-frame analysis uses linguistic knowledge associated with individual lexical items to relate their appearance in canonical syntactic structures to their uses in a semantic environment. It uses a dictionary of case-frames based on the ideas of case grammar originated by Fillmore (1968; see Bruce, 1976 for a general review of case systems). This component uses knowledge about such things as selectional restrictions and the mapping between surface cases (including prepositions) and semantic roles. As seen in Figure 3, the cases for GO are AGENT, TO-PLACE, and DATE.

As we have already observed, interpretation of an utterance must include knowledge of conversational patterns for the appropriate domain. Domain dependent interpretations of utterances were implemented by a simple structure-matching and reconstruction program that operates on case-frames. The example in Figure 3 illustrates how the domain-dependent translation module handles a common conversational pattern for the travel domain: it interprets a statement of desire (the WANT/E) as an instruction to insert the specified event into the trip plan being constructed. In addition, the case frame involving GO is transformed into a description of the TRIP-LEG which is part of the planned trip, with the AGENT of GO becoming the TRAVELLER in the TRIP-LEG and the DATE becoming the TRAVEL-DATE. This simple translation mechanism is obviously very limited; in a more realistic system, the purposes of the client would have to be understood more deeply.

The frame reasoner component of the system was the focus of most of the research and development. It was based on the assumption that large scale structures closely tied to specific procedures for reasoning constitute a framework for producing a mixed initiative dialog system. It uses the frame change description (labelled CMD in Figure 3) to fill in the appropriate information in the trip plan it is building and trigger associated reasoning, as

described later.

The generation of output English is guided by a query-map, a set of templates for all the questions that might be asked by the system. GUS uses a table lookup mechanism to find the appropriate template and generates the English by filling in the template form. This simple generation mechanism is sufficient for the dialog system; generation was not one of the areas of substantial work.

The module that generates questions for the client simultaneously produces one or more skeletons into which his responses can be inserted, if they do not prove to be sentences in their own right. What is being done here is surprisingly simple and works well for most of the fragments we have encountered in response to simple WH-questions. Note that the language generator communicates with the syntactic analyzer using English phrase fragments rather than using a specially constructed formalism. This contrasts with other approaches to the fragment problem, in which the various components of the system are more deeply affected.

THE REASONING COMPONENT

Frames: It is widely believed in artificial intelligence that intelligent processing requires both large and small chunks of knowledge in which individual molecules have their own sub-structure. Minsky's 1975 paper on *frames* discusses the issues and suggests some directions in which to proceed. But, as Minsky stated, his ideas were not refined enough to be a basis for any working system. Our intuitions about the structure of knowledge resemble Minsky's in many ways, and we have appropriated the word *frame*. However, our conceptions are by no means identical to Minsky's, and the two notions should not be confused. The frame structures used in this system were a first step towards a more comprehensive knowledge representation language whose current development is described in Bobrow and Winograd (1977).

Frames are used to represent collections of information at many levels within the system. Some frames describe the sequence of a normal dialog, others represent the attributes of a date, a trip plan, or a traveller. In general, a frame is a data structure potentially containing a *name*, a reference to a *prototype* frame, and a set of *slots*. Frame names are included primarily as a mnemonic device for the system builders and are not involved in any of the reasoning processes. In fact, names are not assigned to any of the temporary frames created during a dialog.

If one frame is the prototype of another, then we say that the second is an *instance* of the first. A prototype serves as a template for its instances. Except for the most abstract frames in the permanent data base, every frame in GUS is an instance of some prototype. Most instances are created during the process of reasoning, although some (for example those representing individual cities) are in the initial data base.

A frame's important substructures and its relations to other frames are defined in its slots. A slot has a *slot-name*, a *filler* or *value*, and possibly a set of attached procedures. The value of a slot may simply be another frame or, in the case of a prototype, it may be a description constraining what may fill the corresponding slot in any instance of the given frame. Figure 4 shows the prototype frame for *date* and the specific date *May 28*, which has no external name. The fact that it is an instance of *date* is indicated by the keyword *ISA* followed by the prototype name.

The *date* prototype illustrates several of the ways in which the values for instance slots can be described. For example, the slot labelled *MONTH* specifies that only a *name* can be used as value; that is, only a literal LISP atom. GUS interprets a standard set of type terms such as *name*, *integer*, *list*, and *string*. The slot for *WEEKDAY* stipulates that a value for that slot must be a member of the list shown in the frame. The slot *DAY* can only be filled by an integer between 1 and 31. The terms *BOUNDED-INTEGER* and *MEMBER* have no special meaning to the interpreter. Any LISP function may occur in this position as a predicate whose value must be non-NIL for any object filling the slot.

Not all of the slots of an instance frame need to be filled in. For example, in *May 28*, only the *MONTH*, and *DAY* are filled in, and not the *WEEKDAY*. A prototype frame provides slots

as placeholders for any data that might be relevant, even though it may not always be present. Only those slot values which are required for the current reasoning process need be put into instances.

```
[DATE
  MONTH      NAME
  DAY        (BOUNDED-INTEGER 1 31)
  YEAR       INTEGER
  WEEKDAY    (MEMBER (SUNDAY MONDAY TUESDAY WEDNESDAY THURSDAY
                   FRIDAY SATURDAY))]
```

a. Prototype for date

```
[ISA DATE
  MONTH      MAY
  DAY        28]
```

b. The instance frame for May 28

Figure 4. Examples of frames

Procedural attachment: We have already referred to *procedural attachment*, a concept first discussed by this name by Winograd (1975), as a central feature of GUS. Procedures are attached to a slot to indicate how certain operations are to be performed which involve either the slot in the given frame or the corresponding slot in its instances. We have found that there are many slots for which some processing is best done by idiosyncratic procedures. For example, there may be special ways of finding fillers for them or for doing other kinds of reasoning about them. This might include verifying that the value in an instance is consistent with other known information or propagating information when the slot value is obtained.

The procedures associated with slots fall into two general classes: *servants* and *demons*. *Demons* are procedures that are activated automatically when a datum is inserted into an instance. *Servants* are procedures that are activated only on demand. The expanded date prototype in Figure 5 contains examples of both classes. On the slot WEEKDAY there is a demon marked by the keyword WHENFILLED and a servant marked by the keyword TOFILL. When a value is filled into the WEEKDAY slot of a date instance, the WHENFILLED statement on the prototype causes the interpreter to invoke the demon FINDDATEFROMDAY. This procedure attempts to compute the appropriate date to fill the other slots in the frame, using the name of the day just entered and contextual information to identify the value uniquely.

The servant GETWEEKDAY on the same slot is only invoked when the name of the week day

is needed. The requirement is satisfied by calling the LISP procedure GETWEEKDAY with the current instance as an implicit argument. The servant attached to the slot YEAR indicates how a default value can be filled in. If the year is given by the client, then this servant will never be activated. However, if the client does not mention the year explicitly, the system will fill in the default value 1975 when any part of the reasoning process calls for it.

```

-----
[DATE
  MONTH   NAME
  DAY     (BOUNDED-INTEGER 1 31)
  YEAR    INTEGER (TOFILL ASSUME 1975)
  WEEKDAY (MEMBER (SUNDAY MONDAY TUESDAY WEDNESDAY THURSDAY
                  FRIDAY SATURDAY))
          (WHENFILLED FINDDATEFROMDAY)
          (TOFILL GETWEEKDAY))
  SUMMARY (OR (LIST MONTH DAY) WEEKDAY)]

```

Figure 5. The frame for date with attached procedures and summary form

The system provides a number of standard servant procedures. ASKCLIENT causes the client to be asked for information that will determine the value of the slot. CREATEINSTANCE indicates that a new instance of a specified prototype should be created and inserted at that location. Some of the values of the newly created frame may be filled in by the procedure, others may be left to be filled through later reasoning or interaction with the client. In addition to standard servants, the builders of the system can program special procedures to compute appropriate values, such as the GETWEEKDAY mentioned earlier.

Summarizing data structures. In Figure 5, the frame for date includes a slot with the special name SUMMARY. A SUMMARY slot appears only in a prototype frame, never in an instance. It gives a format for describing the instances of the prototype to help programmers monitor and debug the system. Thus, instances of date will be described by printing the month and day, e.g. (May 28) or, if they are not known, just the day of the week.

USING FRAMES TO DIRECT THE DIALOG

Frames are used at several levels to direct the course of a conversation. At the top level, GUS assumes that the conversation will be of a known pattern for making trip arrangements. To conduct a dialog, the system first creates an instance of the **dialog** frame outlined in Figure 6. It goes through the slots of this instance attempting to find fillers for them in accordance with the specifications given in the prototype. When a slot is filled by a new instance of a frame, the slots of that instance are filled in the same way. GUS follows this simple depth-first, recursive process, systematically completing work on a given slot before continuing to the next. This is how GUS attempts to retain the initiative in the dialog. Notice, however, that slots may occasionally be filled out of sequence either through information volunteered by the client or by procedures attached to previously encountered slots.

In Figure 6, boldface atoms are frame names, representing pointers to other frames. (Substructures for the frames for **Person**, **Date**, **City**, **PlaceStay**, **TimeRange**, and **Flight** are not shown.) Each of the slots shown in Figure 6 must be filled in during the course of the dialog, usually by invoking a servant attached to the prototype slot. The servants for some slots calculate the desired values from other known data, or (as in the case of frames like **TripSpecification**) simply create a new frame. The servant **ASKCLIENT** obtains information needed to fill a slot by interrogating the client. The default organization of a dialog is determined by the order of the slots which have **ASKCLIENT** as servant, since appropriate questions will be asked if those slots have not been filled by the time they are encountered.

Now let us follow the system as it goes through part of a dialog, with special emphasis on the process of filling in the slots of frames. The dialog and the relevant information about the state of the system are shown in Figure 7. This figure is the beginning of an actual transcript of a session, and the information shown there is provided to allow us (in the role of system builders) to follow the actions of the system.

The dialog starts when GUS outputs a standard message ("Hello. My name is GUS. I can help you plan a simple trip by air."). At this point, GUS knows that it is about to conduct a dialog on travel arrangements, so it creates an instance of the prototype **Dialog** frame shown in Figure 6 and starts to try to fill its slots. (From now on, all numbers in brackets refer to the corresponding lines of the frames of Figure 6. All references to the dialog refer to Figure 7.) The slot **CLIENT** at [1] contains a servant which fills this slot, when necessary, by creating a new instance of **Person**. This is indicated in the first line of the transcript of Figure 7, where the instance of person is shown as {ISA PERSON}. After the slot is filled in, a demon associated with the **CLIENT** slot is triggered, which then puts the same person instance in the **TRAVELLER** slot in [16]. GUS fills the **NOW** slot in [2] by constructing a frame instance for today's date. It then creates a **TripSpecification** instance [3], summarized by **ROUNDTRIP TO ?** in the transcript of Figure 7, to fill the **TOPIC** slot [3].

	Slots	Fillers	Servants	Demons
Dialog				
[1]	CLIENT	Person	Create	Link to TRAVELLER
[2]	NOW	Date	GetDate	
[3]	TOPIC	TripSpecification	Create	
TripSpecification				
[4]	HOMEPORT	City	Default - Palo Alto	
[5]	FOREIGNPORT	City		Link to OUTWARDLEG, AWAYSTAY, INWARDLEG
[6]	OUTWARDLEG	TripLeg	Create	
[7]	AWAYSTAY	PlaceStay		
[8]	INWARDLEG	TripLeg	Create	
TripLeg				
[9]	FROMPLACE	City	FindFrom HOMEPORT	
[10]	TOPLACE	City	AskClient	
[11]	TRAVELDATE	Date	AskClient	
[12]	DEPARTURESPEC	TimeRange	AskClient	Propose-Flight- By- Departure
[13]	ARRIVALSPEC	TimeRange		Propose-Flight- By- Arrival, Link to DEPARTURESPEC
[14]	PROPOSEDFLIGHTS	(SetOf Flight)		
[15]	FLIGHTCHOSEN	Flight	AskClient	
[16]	TRAVELLER	Person	AskClient	

Figure 6. An outline of key frame structures for our dialog

At this point the Dialog frame has been completely filled in so GUS proceeds to fill in the slots of the TripSpecification frame. In [4], a HOMEPORT which is a City is required; GUS assumes, on the basis of an attached servant, that the home port is Palo-Alto. There is no attached servant to find the FOREIGNPORT in [5], so GUS just leaves that slot empty for the moment. When a TripLeg instance is created for the outward leg of the journey, GUS begins trying to fill its slots. A servant for FROMPLACE specifies that it should be filled with the city used for HOMEPORT in the TripSpecification frame, so PaloAlto is filled in. The first slot which has an ASKCLIENT servant is at [10], which requires a city to fill the TOPLACE in the TripLeg, which is the OUTWARDLEG of the TripSpecification [6]. GUS issues the command (CMD) shown at the bottom of Figure 7, which directs the generation of the English question. This is done by a rather elaborate table look up: the result is shown as the last line of Figure 7.

GUS: Hello. My name is GUS. I can help you plan a simple trip by air.

CLIENT = {ISA PERSON} in {ISA DIALOG}
 TODAY = (MAY 15) in {ISA DIALOG}
 TOPIC = (ROUNDRIP TO ?) in {ISA DIALOG}
 HOME-PORT = PALO-ALTO in (ROUNDRIP TO ?)
 FROM-PLACE = PALO-ALTO in (TRIP TO ?)

CMD: (GUSQUERY (DIALOG TOPIC TRIP-SPECIFICATION OUTWARD-LEG TRIP-LEG
 TO-PLACE CITY))

GUS: Where do you want to go?

Figure 7. The beginning of the transcript for the dialog

We continue the trace of the analysis in Figure 8, starting with the client's response to the question. The domain dependent translation contains the information needed to fill the frame slots. The result of the client's English input is that both the `TOPLACE` [10] and the `TRAVELDATE` [11] of the `TripLeg` are filled in.

The system then continues working its way through the entire tree specified by the frames, asking questions of the client. Many of the slots have demons which propagate information to other places in the data structure. For example, when the city that fills the slot `FOREIGNPORT` [5] is found, GUS will insert that same `City` as the place to stay in the `AWAYSTAY` [7]. The `FOREIGNPORT` city also serves as the destination of the `OUTWARDLEG` of the trip and the starting point of the return trip (the `INWARDLEG`). To handle this information, GUS establishes two instances of the frame `TripLeg`, one for the outward leg, the other for the inward leg, and puts the city names in the appropriate slots.

Once a departure specification (some time range before, near or after the desired flight departure) is determined, a demon attached to `DEPARTURESPEC` calls a program which uses this information to propose a flight. Each proposed flight is added to the slot for `PROPOSEDFLIGHTS` [14]. This slot can be used to resolve anaphoric references to flights, based on the order of their mention in the conversation. GUS then tries to determine which of the flights is appropriate to fill in the `FLIGHTCHOSEN` slot [15]. When that has been determined, it will ask for the name of the traveller and confirm the flight.

Many of the slots are marked in such a way that they need not be filled for the dialog to be completed. For example, the arrival specification [13] in each `TripLeg` frame is never requested. This slot is provided as a place to put constraints about the arrival of the flight, if the client volunteers information constraining the desired arrival time. Demons associated with that slot would then be activated to propose a flight based on the arrival time. In a

GUS: What date do you want to return on? ... a query generated by GUS

The context of the next answer is:
 (I WANT TO RETURN ((ON) (*SKIP*))) -- ... The expected context of the query response

CLIENT: On Friday in the evening

CMD: [CLIENTDECLARE ... the domain dependent translation, including context
 (FRAME ISA TRIP-LEG
 (TRAVELLER (PATH DIALOG CLIENT PERSON))
 (TRAVEL-DATE (FRAME ISA DATE
 (WEEKDAY FRIDAY)))
 (DEPARTURE-SPEC (FRAME ISA TIME-RANGE
 (DAY-PART EVENING])

WEEKDAY =FRIDAY in {ISA DATE}
 down when WEEKDAY is put in {ISA DATE} ... triggering a demon to find the Friday's date
 (FINDDATEFROMDAY)
 DAY =30 in (MAY 30)
 DAY-PART =EVENING in {ISA TIME-RANGE} ...evening is interpreted as around 7:30 PM
 DEPARTURE-SPEC =(AT 7 30 PM) in (TRIP TO PALO-ALTO)
 down when DEPARTURE-SPEC is put in (TRIP TO PALO-ALTO)
 (PROPOSE-FLIGHT-BY-DEPARTURE) ... this demon proposes a flight using a departure spec

GUS: Would you like the flight that leaves at 7:45 PM?

CLIENT: That's fine.

Figure 9. Processing a sentence fragment

This sample dialog illustrates how GUS attempts to control a conversation by fitting it to the mold laid down in a structure of related frames. It has a place prepared in this structure for each piece of information that might potentially be used for making travel arrangements. It also has a strategy that will cause the pieces of information that the client must supply to be elicited in a natural order. The sequence of slots in the frames determines the usual course of the conversation, but it will change if, for example, the client volunteers information or asks questions.

REAL AND REALISTIC DIALOGS

There is an important difference between *real* and *realistic* conversations. The simple dialog in Figure 1 is a *realistic* conversation that was actually carried on with GUS. It is much too easy to extrapolate from that conversation a mistaken notion that GUS contained solutions to far more problems than it did. To get an idea of some problems that GUS does not approach, we collected a variety of travel dialogs that clients of a full-fledged system (perhaps the final version of GUS) might expect to conduct. We did this by simulating the system, asking the clients to arrange for round trip air flights between Palo Alto and San Diego, typing all queries and responses on the computer terminal, and pretending that a computer system was interacting with them. In fact, the role of GUS was played by an experimenter sitting at another computer terminal, airline guide, travel books, and calendar in hand, responding to the client.²

² The experimental dialogs were collected by Allen Munro in the LNR research laboratory at the University of California, San Diego.

GUS Do you want a flight leaving at 4:00 PM
 CLIENT Do you have something a little closer to 7
 GUS Do you want the flight at 7:00 PM

a) Interpreting politeness

GUS Do you want the flight arriving at 8:00 PM
 CLIENT When does it leave?
 GUS 6:30 PM
 CLIENT How much?
 GUS \$25.50 round trip

b) Some pronominal reference problems

GUS When would you like to return?
 CLIENT I would like to leave on the following Tuesday, but I have to be back before
 my first class at 9 AM.

c) Giving a reason for flight preference

Figure 10. Fragments of real dialogs, with a person simulating the role of GUS

The two participants -- client and experimenter -- were each seated in independent,

individual sound-isolated experimental booths. They communicated with a special experimental program (designed for tutorial instruction) that presented the experimenter's responses in a block presentation, so it appeared as a realistic approximation of a computer output, without the slow typing rate that would occur otherwise. The system delays were approximately what one would expect for the operation of a complex program (10 to 60 seconds response time).

Some of the problems we found were unexpected. For example, people spent a lot of time telling us about their thought processes and reasons. They made excuses for changing their minds. They hedged a lot about what they wanted. Figure 10a illustrates a type of conversational interaction our current system cannot even begin to handle. When the system proposes a flight at 4 PM, the client requests something *a little closer to 7*. A literal interpretation of that request would be to find a flight that is as close to 4 PM as possible, but in the direction of 7 PM: perhaps the 5:00 PM flight. That, of course, is not at all what was desired by the client. The human experimenter made the natural response of offering the flight that left at 7.

Figure 10b indicates some pronominal reference problems which we did not attack at all. When the client says "when does *it* leave" it is quite obvious that he wants the departure time of the flight referred to in the previous sentence. For his question "how much," a response that "all of the plane leaves" seems somewhat inappropriate. In this case, the client is not referring to the previous system response, but rather is asking about the cost of the flight. But a response such as "how much" can sometimes refer to the previous system response. Suppose the system had just stated "They serve food on that flight." In this case, the client's query could be appropriately interpreted by the system as referring to the quantity of food. GUS cannot solve the problem of determining when a response is meant to refer to the previous question and when it is not.

Figure 10c illustrates how people provide extra information about their motivations. In a system with a better model of human needs and desires, this would be useful for suggesting alternatives that might otherwise be ruled out.

CONCLUSION

Computer programs in general, and programs intended to model human performance in particular, suffer from an almost intolerable delicacy. If their users depart from the behavior expected of them in the minutest detail, or if apparently insignificant adjustments are made in their structure, their performance does not usually change commensurately. Instead, they turn to simulating gross aphasia or death. The hope, which has been at least partially realized in GUS, is that the notions of procedural attachment and scheduling, as well as being realistic cognitive models, will make for more robust systems. We were pleased, for example, by the way the system's expectations could evolve in the course of a single conversation. The client would occasionally seize the initiative, volunteering information that was not asked for or refusing to answer a question as asked and GUS was able to respond appropriately in many cases. It would be misleading to press these claims too far. GUS never reached the stage where it could be turned loose on a completely naive client, however

cooperative. But, to one familiar with other systems of the same general kind, the impression of increased robustness is clear.

GUS represents a beginning step towards the construction of an intelligent language understanding system. GUS itself is not very intelligent, but it does illustrate what we believe to be essential components of such a system. An intelligent language understander must have a high quality parser, a reasoning component, and a well structured data base of knowledge. The knowledge is of several types, from language specific information and expertise in the topic areas in which it can converse to broad general knowledge of the world that must be used to interpret people's utterances. This knowledge tends to be taken for granted by most native speakers of the language, hence often left for the listener to infer. The system must be capable of giving direction to the conversation, but it must also be flexible enough to respond to novel directions set by the clients. The system must be able to make use of a large external data base and to understand what information must be retrieved and processed in depth. There must be an intimate connection between its representation of structural knowledge and the procedures used to process knowledge. A general framework for representing knowledge must be able to encompass all the different necessary forms of knowledge. In our future studies of GUS, we intend to broaden the general framework for representing knowledge, as well as to increase the power of the components of the system. Preliminary steps in this direction include the development of improved systems for language analysis (Kay & Kaplan, 1976) and a knowledge representation language (KRL: Bobrow & Winograd, 1976).

References

- Bobrow, D. G. & Collins, A. M. (Eds.) *Representation and Understanding: Studies in Cognitive Science*. New York: Academic Press, 1975.
- Bobrow, D. G. & Wegbreit, B., A model and stack implementation of multiple environments, *Communications of the ACM*, 1973, 16, 591-603
- Bobrow, D. G. & Winograd, T. An overview of KRL, a Knowledge Representation Language. *Cognitive Science*. Vol 1. No 1. 1977
- Bruce, B., Case systems for natural language. *Artificial Intelligence*, 1975, 6, 327-360.
- Carbonell, J. R. AI in CAI: An artificial intelligence approach to computer-aided instruction. *IEEE Transactions on Man-Machine Systems*, 1970, MMS-11, 190-202.
- Carbonell, J. R. Mixed-initiative man-computer instructional dialogues. Unpublished Ph.D. dissertation. Cambridge, Mass: Massachusetts Institute of technology, 1970.
- Dahl, O. J., & Nygaard, K., SIMULA--an ALGOL-Based Simulation Language, *Communications of the ACM*, 1966, 9, 671-678.
- Fillmore, C. The case for case. In E. Bach and R. T. Harms (Eds.), *Universals in Linguistic Theory*. New York: Holt, 1968.
- Goldberg, A. & Kay, A. (Eds.) SMALLTALK-72 instruction manual. Xerox Palo Alto Research Center SSL-76-6. Palo Alto, Ca. 1976
- Gordon D., & Lakoff, G., Conversational postulates, Papers from Seventh Regional Meeting, Chicago Linguistic Society, Chicago: University of Chicago Linguistics Department, 1972.
- Grice, H. P. Logic and conversation. In P. Cole & J. L. Morgan (Eds.), *Studies in Syntax*, Volume III. New York: Seminar Press, 1975.
- Kaplan, R. A general syntactic processor. In R. Rustin (Ed.), *Natural language processing*. New York: Algorithmics Press, 1973a.
- Kaplan, R. A multi-processing approach to natural language. *Proceedings of the 1973 National Computer Conference*. Montvale, N.J: AFIPS Press, 1973b.
- Kaplan, R. On process models for sentence analysis. In Norman, D. A., Rumelhart, D. E., and the LNR Research Group. *Explorations in cognition*, San Francisco: Freeman, 1975.

Kay, M. The MIND system. In R. Rustin (Ed.) *Natural language processing*. New York: Algorithmics Press, 1973.

Kay, M. & Kaplan, R. Word recognition. Palo Alto, California: Xerox Palo Alto Research Center, 1976.

Minsky, M. A framework for representing knowledge. In P. Winston (Ed.), *The psychology of computer vision*. New York: McGraw-Hill, 1975.

Reddy, D. R., Erman, L. D., Fennell, R. D., & Neely, R. B. HEARSAY speech understanding system: An example of the recognition process. *Proceedings of the Third International Joint Conference on Artificial Intelligence*, Stanford University, August 1973.

Norman, D. A., Rumelhart, D. E. and the LNR Research Group, *Explorations in cognition*. San Francisco: Freeman, 1975.

Rovner, P., Nash-Webber, B., & Woods, W. A. Control concepts in a speech understanding system. *Proceedings of the IEEE Symposium on Speech Recognition*, Carnegie-Mellon University, April 1974.

Simon, H. *Sciences of the artificial*. Cambridge: Massachusetts Institute of Technology Press, 1969.

Teitelman, W. INTERLISP reference manual. Palo Alto, California: Xerox Palo Alto Research Center, December, 1975.

Walker, D., Paxton W., Robinson, J., Hendrix, G., Deutsch, B., Robinson, A. Speech understanding research. Annual report, Project 3804. Artificial Intelligence Center. Stanford Research Institute, 1975.

Winograd, T. Frames and the declarative-procedural controversy, In D. G. Bobrow and A. M. Collins (Eds.), *Representation and Understanding*. New York: Academic Press, 1975.

Woods, W. A. Motivation and overview of BBN SPEECHLIS: An experimental prototype for speech understanding research. *Proceeding of the IEEE Symposium on Speech Recognition*, Carnegie-Mellon University, April 1974.

However, such monitors currently cost \$3K to \$6K. Higher *line-rate* monitors using these CRTs are also becoming available, but their cost is currently >\$10K. All color monitors have some misconvergence, especially near the edges of the screen. This annoyance is often quite noticeable on text or other sharp edges. Trinitrons have color stripes instead of dots and thus exhibit less convergence error, but are lower resolution than color dot CRTs. Flicker can be quite noticeable on color CRTs at standard 30 frame/sec rate, particularly with text and small detail. Slower color phosphors are not readily available.

2. Alternatives.

There are several major alternatives to consider for a color display capability on the Alto:

2.1 With a reasonable amount of effort.

2.1.1 As the *primary* (working) display, *compatible* (i.e., 875-line, 3:4 aspect). This alternative is the simplest, but only 2 colors are possible due to bandwidth limitations. However, by judiciously loading the color table, *different* pairs of 2 colors can be displayed in different bands on the screen (See color Alto document). Monitor very expensive. Convergence and flicker may be quite annoying for day-to-day use.

2.1.2 As the *primary* (working) display, *incompatible* (conventional 525-line is the only reasonable alternative). This was the color Alto solution described above. $480 \times 640 = 300K$ pixels as compared to $808 \times 606 = 480K$ pixels with the standard Alto display. Compatible in the sense that scan lines in the color Alto are *longer* (640 vs. 606) so that most display bit maps will display properly (operating system exec, Bravo, etc.) and the displayed area will just be shorter in the vertical direction. 2 bits/pt (4 colors) possible at full resolution. Convergence and flicker.

2.1.3 As a *secondary* (additional) display. If the standard Alto display remains available, then questions of software compatibility are avoided. Both displays could not be on at the same time, however, due to bandwidth limitations.

2.2 More radical schemes. Other possibilities include: increasing the memory bandwidth via a new memory card or memory bus design, plugging in some new memory which contains the display bit map, encoding color pictures (run or area coding) and accessing via a new display controller, encoding color information separately and combining it with the bit stream from the standard display controller, etc.

Inter-Office Memorandum

To ALTO II Users Date February 23, 1978

From Doug Stewart Location El Segundo

Subject Second Disk Drive For ALTO Organization ED/SPG

XEROX

Filed on: [XEOS]<Ludolph>Hardware>SecondDisk.press

The following is a list of all of the equipment required to put a second Diablo Model 31 disk drive on an ALTO system. All may be purchased directly from Diablo.

Item	Qty. Req'd	Description
1	1	Model 31 disk drive, P/N 15532-06, with: Option -004, 1562 KBS Option -019, Extended format
2	1	Power Cable, P/N 11188
3	1	Cable, Diablo P/N 11245-xx, where xx is the length in inches (should be about 60 inches if drive is to sit on top of ALTO cabinet).

c: Frank Ludolph

Whole ALTO World Newsletter

Technology and Tools

XEROX

January 31, 1978

SPECIAL ANNOUNCEMENTS

FINAL ALTO II SPARES BUILD - SPG is now taking orders for the final Alto II Spares Build. This is for Alto II boards excluding accessories such as Orbit and Trident. *The closing date for this build is March 1, 1978.* Contact Terry Haney, SPG.

WHOLE ALTO WORLD MEETING - The next Whole Alto World meeting is scheduled to be held from 9AM to 3:30PM February 7, 1978, in the Green Room at XEOS, Pasadena. Liz Bond is our hostess. The topics discussed will include maintenance, build activity, protection of intellectual property, software, and a demonstration of Smalltalk.

GENERAL NOTES

SUBSYSTEMS CATALOG - The Subsystems Catalog has been revised to include new subsystems and a functional cross reference. The latter aids in locating a program to perform a given task. When a program has been located, the alphabetical listing can be used to obtain a more complete description and direction to the documentation.

ALTO NETWORK DIAGRAM - This diagram, illustrating server location by Ethernet, has been revised to include the servers' names and addresses. A copy is attached.

TOOLS

HARDWARE

ALTO II WORKSTATION MULTIPLEXER ADAPTER - An Alto II Keyboard Buffer is now available that permits the Workstation Multiplexer to be used with Alto II keyboards. (The multiplexer itself is used to build a 4X4 arrangement of Altos and workstations such that any workstation may be manually switched to any Alto for exclusive use.) Cost of the keyboard buffer is \$275 plus \$125 per workstation interface (one per Alto II workstation). Also required are cables and the multiplexer (total cost is \$2.5K to \$3K depending on exact configuration).

TRIDENT MULTIPLEXER - The following was excerpted from a message by Ron Freeman, SPG:

The 74161s used for the sector counters on the Trident Multiplexer boards are being used improperly resulting in erroneous operation of the sector counters unless the 74161 happens to be manufactured by National Semiconductor...

As this problem does not exist in the design of the 74LS161, an E.O. is being written to use the LS device for this design.

Several of these modules have been delivered and installed in Alto IIs, all apparently with the National part used. You can suit yourselves as to the desirability of replacing the 74161s on these units; however, should they be returned to SPG for repair in the future, the 74161s will be replaced at that time with the 74LS161s.

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local IVY server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [XEOS].

ReReleases - Subsystems

CHAT - Minor fixes have been made and a curve displaying protocol added. Retrieve <Alto>Chat.run. The documentation is unchanged.

FIND - Enhancements to this subsystem include switches to distinguish upper and lower case, to print the entire Bravo paragraph containing a match, and to print all text between blank lines containing a match. Retrieve <Alto>Find.run. Updated documentation is available from <AltoDocs>Find.tty.

FRED - The restriction limiting the range of character codes to 7-bit ASCII has been lifted. Load <Alto>Fred.dm. The documentation is unchanged.

MADTEST - The nature of the changes are unknown to me. This subsystem is available from the bootserver or retrieve <Alto>Madtest.run. There is no documentation.

OEDIT - This subsystem has been extended to provide a simple search command and to display a range of values for rapid browsing. Retrieve <Alto>Oedit.run. New documentation is on <AltoDocs>Oedit.tty.

SCAVENGER - Changes involve improved DiskDescriptor reconstruction. Retrieve <Alto>Scavenger.tty. The documentation is unchanged.

SETTIME - If this subsystem is unable to obtain the date and time from a time server, it will now display an appropriate message and prompt you to type in the date and time. It has also been altered for the new time standard. Retrieve <Alto>SetTime.run. The updated documentation is on <AltoDocs>SetTime.tty.

TYPE - This version fixes the bug that caused long lines to scroll off of the top of the screen. Retrieve <Alto>Type.run. The documentation is unchanged.

ReReleases - Packages

ETHERBOOT - This new version conforms to the new time standard, can direct its boot-load request to a specific address, and can take alternative action if a boot-load request fails to be honored. The documentation, <AltoDocs>EtherBoot.tty, has been updated.

GP - This package for parsing command lines deletes the DefaultArg routine and slightly alters the ReadParam routine. The revised documentaion may be retrieved from <AltoDocs>GP.tty.

DOCUMENTATION DISCREPANCY - It has been reported that there is a discrepancy between the documentation and implementation of the UtilStr package. For the routine Db1Sub, the parameters Db1Minuend and Db1Subtrahend are reversed. Also, the Db1Div routine returns the remainder in the first word and quotient in the second, not a 32-bit quotient as some have understood the documentation to say.

TECHNOLOGY

The topic of this month's paper is the emergence of an applied psychology that can be used by designers to develop user interfaces, to predict user performance on various design alternatives, and, in some cases, to compare it to the theoretical limit. While a substantial theory does not yet exist, the authors present a view of what an applied psychology might be like and assert that a sufficient amount of knowledge exists now to begin impacting the human side of Office Information Systems.

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC]<AltoDocs>WAWnews.press or may be obtained from the editor, Frank Ludolph, XEOS, by messaging <Ludolph> or calling Intelnet 8*923-4356.

Applied Information-Processing Psychology for Xerox Office Information Systems

Stuart K. Card
Thomas P. Moran
Allen Newell*

AIP Memo 99
July 1977

XEROX

Systems Sciences Laboratory
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, California 94304

* The authors constitute the staff of the Applied Information Processing Psychology Project of the Systems Sciences Laboratory at the Palo Alto Research Center. A. Newell is a consultant, regularly associated with Carnegie-Mellon University. The order of the authors above is alphabetical. There is no administratively designated leader of the project.

SUMMARY

This essay addresses the potential contribution of an applied psychology to developing Xerox's Office Information Systems. Three years ago, under the impact of a cognitive psychology that finally seemed to be developing an adequate theoretical base, the Applied Information Processing Psychology Project started down the path to creating a new applied psychology that would make use of that base. This essay is a restatement of the original vision in the light of our attempts in the interval since then.

An important component of any office information system is its interface to the user. We envision an applied science of the user that: (1) is based on an information-processing theory of the user, (2) permits both quick and detailed calculations of user behavior, (3) contributes to the evaluation of proposed system designs as well as to the generation of new designs, and (4) is applied directly by the system designers themselves and not just by professional psychologists.

A model of the user consists of the goals the user sets for himself and the methods he chooses to attain those goals. The goals reflect the task to be accomplished and the methods reflect the available means, including the office system. A method consists of a conditional sequence of sub-goals and operations, which are behavior patterns that occur across many different task situations and for which empirical data is collected. Such a model allows user behavior in a new task to be predicted by postulating the goals and methods required in the new task. For example, we have made a prediction (for SDD) of the time to format the pages of a document with a reasonable (but as yet non-existent) document composition system, and this was done using our models and data from text editing. Another example of calculation from theory is the time it takes to point to text on a display with a "mouse". The movement time of the mouse is accurately predicted by Fitts' Law; and comparison with other movement studies reveals that the mouse moves as fast as a human can control and thus that no other pointing device will be faster.

Looking to the future, we can imagine the shape an applied psychology might take. The model of the user could be packaged as a Simulated User, a computer program that produces statistically appropriate user behavior and that can be combined with an office system simulation program. Another application is a notation for specifying system command languages. The notation would embody the User's Model, i.e., the conceptual model the user has of the office system. Such a notation would allow the designer to design the system explicitly in terms of this User's Model. This illustrates how we see an applied psychology being brought directly into the design process. A more general design tool would be a Handbook of Applied Cognitive Psychology, organized around classes of design problems and showing how to bring psychological results to bear on them.

We believe that such an applied psychology is possible. The question is how to pursue it. The AIP group's effort to date has been a mixture of in-depth studies of basic issues to build the theory and of short application studies to test the theory. Another element of our strategy is to explore the domain of office task situations, each new task explored becomes a test of the theory so far and an extension of our knowledge. As can be seen in the table at the end of this essay, our effort thus far has concentrated on the manuscript editing task. This has allowed us to pursue some fundamental issues of how to build performance models. There are many other basic issues -- such as, why do users make errors? and how does one learn a new system? -- that need to be addressed; and there are many other task areas to explore. Our view is that we have made progress over the last three years and that we are on the right track to attaining our goal.

Applied Information Processing Psychology for Xerox Office Information Systems

Stuart K. Card, Thomas P. Moran, & Allen Newell

This essay addresses the potential contribution of an applied psychology to developing Xerox's Office Information Systems. Three years ago, under the impact of a cognitive psychology that finally seemed to be developing an adequate theoretical base, the Applied Information Processing Psychology Project (AIP for short) started down the path to creating a new applied psychology that would make use of that base. This essay is a restatement of the original vision in the light of our attempts in the interval since then.

The Problem and the Vision

Within the world of Xerox Office Information Systems, some propositions are almost self-evident:

- A. The performance of an office system depends strongly on its being easy to learn and use.
- B. Because the users of office systems are not technically oriented, the importance of the interface between an office system and its users is magnified.
- C. The marketability of an office system depends on the ease, grace, and apparent simplicity of its use, even beyond its actual performance once firmly incorporated into a customer's operation.
- D. The style of information systems most favored by Xerox -- namely, highly interactive, decentralized, personal computing -- is exactly the one that most emphasizes the human-system interface.

These propositions accurately reflect the large potential contribution to be made through good design of the human-system interface. Further argument on this point seems unnecessary.

This does not establish automatically that an applied psychology is needed for good

interface design. Interface design is still largely an art. Sensitive designers often produce elegant interfaces, and the iteration of designs in response to user reactions can remove rough edges. But an applied science of how humans use office systems -- how they think, plan, follow procedures, perceive, communicate -- should have much to say about the design of the interface. That any room for doubt exists about an applied psychology's contribution reflects the variability and complexity of human behavior. It also reflects psychology's still immature state compared with the physical (and more recently the biological) sciences. But all sciences grow, and psychology's potential for substantial contribution to an applied area, such as office information systems, must be continually reassessed.

There has indeed been growth in psychology, especially in cognitive psychology, which has come to view man as an active processor of information who can represent his environment symbolically, create goals to express his desires, and engage in strategies of action to attain these goals. The key to treating the information processing aspects of man scientifically and quantitatively has been the developing understanding of man-made information processing systems, i.e., computer and control systems.

This cognitive psychology provides us with a vision of an applied psychology of office systems that is mundane by the standards of physical engineering and science, but novel in the extreme for psychology. It almost catches the breath:

1. *Theoretical base:* There will be a unified model of the office person as he interacts with office information systems; this framework will allow the progressive cumulation of both facts and theory.
2. *Calculations:* The theory will permit calculations of results, especially of the "back-of-the-envelope" type so dear to the practicing engineer and designer; better approximations may be had by applying more effort.
3. *Scope of application:* Contributions will occur across the full range of the design process, from the evaluation of office systems, both prototypes and proposals, to the generation of new design conceptions.
4. *Integration into design practice:* The results will become tools of the working designer, not just the professional psychologist, thus maximizing the opportunities for application.

This is a vision, not a present reality. Visions are useful if they are close enough to reality for current finite efforts to make perceptible headway toward them. Enough ingredients and demonstrations of power have been provided by recent advances in psychology to create a suitable base, though much in this vision is still foreign to traditional techniques (back-of-the-envelope calculations, for example) and must be

forged on site. Still, it was this vision, and the assessment of its attainability, that underlay the initiation of AIP at PARC three years ago. It is the perceptible headway that we've made in the three years that leads us now to a restatement.

The next section will make concrete what we mean by an applied psychology that is quantitative and admits of calculation, yet deals in goals and symbolic behavior. Then we will project some of the ways that we see such an applied psychology operating when it is mature. This will lead to a brief assessment of where AIP currently stands in creating that mature science, and what needs to be done to move along that path.

The Shape of an Applied Psychology

The Model of the User

Central to the enterprise is a model of the human user and how he interacts with office systems. This model comes from a wide range of basic experiments in cognitive psychology that have gradually let us piece together a reasonable picture of the functional information processing architecture of the human.

The simplest version of the model takes the user to have a single long term memory of unlimited size which holds essentially permanent knowledge. There is also a working memory of very limited size which holds the knowledge of the current task context; this represents the user's focus of attention. What the user does at each given instant of time is determined by the contents of this working memory. Knowledge flows into the working memory from both the external environment of the user and from his long term memory. Both of these flows are determined or moderated by the knowledge in the working memory at each instant. Similarly, the user's initiation and control of external (motor) actions and of the addition of new knowledge to his long term memory is determined by the contents of working memory.

Some important general limitations of the user of office systems can be derived directly from this simple view of the human information processing architecture:

- (1) The user can do only one main thing at a time -- what the working memory determines at each instant.
- (2) He can work only so fast -- there is an upper bound to the cycle time of the system (in the range of 10 to 20 cycles per second).
- (3) His working memory is limited (less than a dozen items of knowledge) and overloading the working memory causes some immediate knowledge to be lost.
- (4) His knowledge in long term memory can only be retrieved if it can be accessed via the knowledge already in working memory. Thus, complete failures to recall relevant knowledge are possible, as are retrievals of inappropriate knowledge because the retrieval clues were inadequate.

- (5) Adding knowledge to his long term memory takes much longer than retrieving knowledge (knowledge can be accessed in a single cycle, less than a tenth of a second, but takes several seconds to be stored away). The user is forced to rely on his limited working memory to cope with rapidly changing task data.
- (6) The information that the user encounters in the outside world must be encoded if it is to become usable; and this can be done only in terms of the knowledge already available in the long term memory. Thus, the user can only learn new things in terms of knowledge he already has.

Calculating from Theory: Goals and Methods

To be able to calculate with a theory, one must be able to represent the behavior of the system in terms of some primitive components and how they are composed. Editing manuscripts with interactive computer editing systems provides a good example. It is typical of the operations that will occur in future office systems. One important quantity is the rate at which manuscripts are edited. The editing task can be analyzed into a sequence of "unit tasks", which are relatively independent subtasks (e.g., locating in the manuscript the next correction to make and then making it). According to the theory, the behavior of the user in each unit task can be analyzed into a hierarchy of goals and methods that resolve, at some level of detail, into a sequence of operations. For simple calculations, these operations can be assumed to take constant time.

An example of such an analysis is shown below for an editor called Poet. The operations are indicated by CAPITALS, and their average times (empirically determined) are given in the right column:

Goal: edit unit task	
Goal: get unit task from manuscript	
GET-NEXT-PAGE (of manuscript),	2.1 seconds
if at end of current page	
GET-UNIT-TASK (from manuscript)	1.9 seconds
Goal: carry out unit task obtained	
Goal: locate line in file containing unit task,	
if not already on the current line	
Choose one of the following:	
USE-SEARCH-STRING-METHOD	3.9 seconds
USE-NEXT-LINE-METHOD	4.3 seconds
Goal: alter text according to unit task	
Choose one of the following:	
USE-SUBSTITUTE-COMMAND	3.6 seconds
USE-MODIFY-COMMAND	9.7 seconds
VERIFY-EDIT	1.5 seconds

Though written out in English phrases, this is a formula. It specifies the behavior of the user during a unit task in terms of some basic components, and permits calculation of its duration. As shown, it contains options for the user that are determined by specific details on the manuscript. For example, selecting whether to locate the next line to be edited by using the NEXT-LINE method (hitting the NEXT-LINE key successively) or the SEARCH-STRING method (having the system search for a specific string of text) depends on specifics of the text. Empirical rules are known for how users make such choices, e.g., use the NEXT-LINE method if the next unit task is less than five lines away from the current unit task line.

The parts of the formula can be expanded into a finer level of detail. For instance, the operation USE-SUBSTITUTE-COMMAND can be converted into a goal and then expanded into smaller operations:

Goal: use Substitute command	
SPECIFY-COMMAND	1.5 seconds
SPECIFY-ARGUMENT (old text)	1.4 seconds
SPECIFY-ARGUMENT (new text)	1.4 seconds

These operations can be expanded yet further. But already the analysis has been driven down to the level of operations that are quite independent of the Poet editor, for many systems are composed of similar command and argument specifications.

The use of such a formalism can be illustrated by an opportunity that fell our way. It became of some importance within the System Development Division to assess the rough cost of formatting a page of text by means of an interactive computer system, i.e., assembling text, figures, and footnotes into the final physical arrangement on a page. This posed a typical design quandry. No such systems exist, so there is no directly relevant operating experience to use as a guide. Yet some gross estimate is needed to guide the design process into considering alternative product schemes of quite different character.

Approached by Harold Hall of SDD, we tried to see what our applied psychology in its current partial state could do in the way of a quick calculation. We analyzed the page formatting task into unit tasks. Examination of typical pages in reports yielded rough frequencies for the occurrence of the different kinds of unit tasks. We then analyzed how the user would perform each unit task, creating formulae analogous to the ones above and assuming a computer-formatting system with plausible properties. Enough contact could be made with the operations in the systems we have studied, so that numerical estimates of the duration of each unit task could be calculated. Taking these together with the frequencies of occurrence yielded a total estimated time to format a typical page. We also investigated the non-additivity of the unit tasks (since related tasks done together avoid repeating common operations) and showed that the correction was probably too small to make any difference.

The estimate obtained is surely in error. However, it is probably substantially better than the more intuitive estimates the designers were drawing from their own experience. It cannot be checked directly, since page-formatting systems do not exist. We did create one quick analog in the laboratory and took measurements of actual performance to provide some gross check. We could not quite confine the analysis to the back of an envelope; it took about a week. Even so, that time scale is well matched to the demands of the actual design process it was attempting to aid.

Calculating from Theory: Pointing to Text on a Display

Another example of a calculational applied psychology comes from the problem of how users point to text on computer displays. There are many different devices for moving the cursor from one point on the screen to another, and the usual range of opinion exists about the advantages and disadvantages of different devices. This is a classic human factors situation where the specific devices would be evaluated by running experiments under a range of conditions (some evaluations of cursor movement devices exist in the literature).

One can do better than this, however. According to a law proposed by Paul Fitts in 1954, the time it takes a skilled user to move his arm in a task demanding accurate placement is given by

$$t = k_0 + k \log_2(d/s + .5) ,$$

where d is the distance to be moved and s is the size of the target region into which the movement must be made. The smaller s is, the more accuracy is demanded. The formula says that what counts is only the relative accuracy (d/s). Further, the time increases only as the logarithm of this accuracy (the number of bits in the discrimination). The constants, k_0 and k , reflect the movement situation and the skill of the operator. Fitts' Law is one of the most robust results in psychology, and it has been verified in a wide range of experiments.

The usefulness of Fitts' Law can be seen by considering another question of current interest to SDD: how good is the "mouse" as a cursor control device, and are there any better alternative devices? Fitts' Law can be applied to the case of cursor movement devices by taking d to be the distance on the screen that the cursor has to move and taking s to be the size of the target text. Data from a series of experiments that we ran with the mouse give the following fit to Fitts' Law:

$$t_{mouse} = 1.03 + .096 \log_2(d/s + .5) \text{ seconds} .$$

One more step is required to understand the implication of this result. In the literature on motor control, the value of .1 seconds per bit for the constant k shows up repeatedly in different arrangements as a lower bound, i.e., as the fastest the human can operate. This appears to be a limit due to the information processing necessary to guide the motion.

The formula for the mouse above not only tells us how fast it is for different targets, but also it tells us that it is unlikely that other continuous-movement devices can be found that improve much upon it. (Even pointing directly to the screen with a finger would not be much faster!) What hope there is lies in trimming the one second set-up constant, k_0 , which involves the basic reaction needed to respond to the occurrence of the task itself, no matter what device is used, and the pushing of the mouse button. In fact, we have tested other pointing devices in our laboratory, and none are faster than the mouse.

The Yield of an Applied Psychology

The previous section attempted to give some flavor of the applied psychology based on viewing the user as an information processing system. Drawing on present work, we showed how one might calculate approximate answers to quantities that are of applied interest in office information systems. Here we would like to reach beyond what we now feel sure about to give some examples of ways we envision applications occurring.

Simulated Users

System simulation is a standard design technique in computer science. It is used to evaluate proposed designs before they actually exist and to permit explorations of design variations without the expense and time to construct alternative prototype systems. Simulations always work with some simulated model of the external environment. In some areas, such as standard computer time sharing systems, rather simple models of the environment are adequate. However, for the office information systems envisioned by Xerox, the interaction between the user and the systems is too intimate for any simple model to suffice.

A "Simulated User" is a program that will produce the behavior of a human user in conjunction with a simulation of a proposed system. It will generate sequences of tasks to do according to some probabilistic model of the task domain, and for each task it will generate commands to instruct the computer system to carry out these tasks. It will choose methods based on the same principles that appear to govern human choices, and it will make errors with the same relative frequencies. A Simulated User can be tuned to show certain kinds of user characteristics (e.g., to be a particularly sloppy user or a highly efficient one); and it can behave at any level of detail, depending on how it is to be used.

A Simulated User is a way of packaging psychological knowledge that makes it fit directly into the design tools of the system designer. Simply by using it in conjunction with his simulation he brings to bear relevant psychological effects in a direct and continuous way. Furthermore, the Simulated User is cast in exactly the terms that a practicing software or hardware designer can understand; he can explore its behavior when it turns up unexpected issues in the design. It will not be a black box.

A Design Representation for Command Languages

The artistic side of design lies largely in the generation of ideas and the scientific side lies largely in their testing. It is easier to imagine how to use scientific knowledge of the user to evaluate designs than to help in their generation. Yet a good applied psychology must do both.

Common to all interactive computer systems is the command language -- the language used by the user for specifying to the computer what he wants it to do. Their design is an important aspect of the design of interactive office systems. It is almost completely an intuitive art currently, with its inevitable collection of lore. One notion is the "User's Model": that the user develops a conceptual picture of the interactive system that serves to guide his actions and expectations of how the system will behave. Whether a system is simple and elegant or awkward and inconsistent is a reflection of its implicit User's Model. A simple User's Model reflects in ease of learning, though it is recognized that any finite system can eventually be learned to permit expert performance. Unfortunately, no scientific substance has ever been given to the notion of a User's Model, and it remains essentially a mythical beast.

Imagine, then, a notation for specifying command languages with two main components. The first is a notation for specifying the computer system being controlled by the user via the command language. This specifies the user's view -- the entities that the user understands, their relations to each other, and the types of actions the user sees the computer as being able to perform. In short, this is the User's Model. The actual implementation of the system must stay within the limits imposed by the User's Model -- or the user's view will actually be inconsistent -- but otherwise it is quite free.

The second component of the notation is a grammar for the command language itself, defined in terms of linguistic constructs -- commands, arguments, contexts, state variables -- and in terms of how it designates and evokes the elements of the User's Model. The user sees his commands as designating to the system various elements as defined in the User's Model. The grammar can specify command languages of arbitrary complexity and complexion; it encompasses all possibilities for command languages, given a User's Model.

By providing a separate representation for the User's Model and for the command language per se, this scheme permits the User's Model to be designed explicitly, rather than to provide only informal heuristic guidance. This wholly changes the space in which user-system interfaces are designed.

A command language representation embodying a User's Model will not look to the practicing designer like a psychological theory. He will see it simply as a symbolic tool that lets him write down aspects of proposed systems -- from sketches to completed specifications. Nevertheless, the representation will be a psychological theory. Experiments will have verified that the representation reflects how humans actually view

office systems. For example, users' learning of systems will be describable in terms of acquisition of the User's Model. Doing what comes naturally within the representation will lead to designs that take human cognitive structure into account.

The Style of an Applied Psychology

Questions concerning the user occur everywhere in the design of office information systems. Often they are not the central question, but operate more as a check against solutions being user-foolish. Being pervasive, these applications will be made by the designers themselves or they will not occur at all. An important component of our vision, then, is to get the tools into the hands and heads of the practicing designer. We do not thereby believe the designer, by training a software or hardware professional, is to become also a psychologist. Gaps in expertise always exists between disciplines. But the main result of an applied psychology will be measured by its lifting the whole design process in its consideration of the human interface, not by a few psychologically deep applications, made by the psychologists themselves (though we hope for those too).

The examples we have given already show some ways in which this transfer will occur. The Simulated User and the command language representation (with the User's Model) both provide tools that are used directly by the designer and incorporate psychological knowledge in their very structure. More generally, the psychological theory we've illustrated is cast in the same language as that used by the engineer, so he can assimilate it easily.

Perhaps a small example will show what we mean. Recently we gave a seminar about the pointer studies and Fitts' Law. A few hours afterwards, Lynn Conway, the Manager of the LSI Systems Group in SSL, came to us with a computation based on these results (and in fact done on the back of an envelope) that showed how to decide the placement of some active touch-areas on the display of an interactive, computer-aided, integrated circuit design program they were at that moment developing. It was only the smallest of applications, yet it has exactly the right flavor of spontaneous transmission to the working designer.

The incident with Lynn Conway was serendipitous. The transfer of expertise will actually be more deliberate. One product of AIP will be a Handbook of Applied Cognitive Psychology, which will not only provide systematic psychological data, but will be organized around classes of applied problems that can be solved by calculation from the included data. This Handbook will be deliberately aimed at the engineer-designer. Beyond this there will exist arrangements for designers to work with AIP for short periods on areas of their own special concern, where the transfer of expertise will occur rapidly and effectively.

What Needs to be Done

In our own view, the efforts of the first three years have reinforced the premise on which AIP is based: a useful applied psychology is possible. In particular, it is possible for Xerox office information systems. The question, in our eyes, is how to pursue that goal.

As we have tried to make clear, AIP's fundamental product is the capability to apply psychology throughout the entire domain of interactive information systems. This domain is populated by a great diversity of task situations, almost none of which have ever received even the slightest psychological analysis. One dimension of AIP's work then is to explore this total territory. Each new task domain becomes simultaneously an extension of our knowledge and a test for how it has cumulated so far.

The work of AIP to date can be viewed as an initial foray into the domain of interactive systems. This view is presented in the table at the end of this essay. It shows completed, ongoing, and planned AIP studies. The vertical dimension of the table lays out the task areas and the studies in those areas. The horizontal dimension lays out the research aspects of each study, both the *basic ingredients* that it contributes to the applied science and the types of *applications* it provides. The ingredients for tackling each area consist of: (1) *exploration* of the area -- the acquisition of a first-order view of the nature of human action in the area and how it fits (or does not fit) within existing psychological science, (2) creation of successive orders of *theory*, (3) each coupled with experimental *verification*, (4) along with the construction of a *data base* of facts and principles, which puts this knowledge into accessible form. These ingredients often do not come along without the construction of corresponding *tools*. Finally, we distinguish three types of applications: (1) quick *calculation*, (2) quantitative *evaluation*, (3) and *use in design*.

Given the table, one fundamental strategy of AIP can be easily described: engage in an iterated series of attempts to fill out the table and to expand it along both the dimensions of tasks and types of applications. Commitment to a unified theory of human information processing and to cumulation of results implies that each study must contribute to the whole. Commitment to finding first-order effects and limited-effort calculational techniques implies that one task area cannot be developed too deeply before others become too inviting. No time dimension exists in the table, since each study should be short enough to make its contribution and permit another iteration to commence. Commitment to embedding applied psychology as a workaday tool of the professional systems designer puts a premium on learning how to get into a new application area, explore it, create a local theory for the phenomena, collect a minimum of data, get the results, and get out -- all in a big hurry!

The domain of interactive man-machine systems, like most task areas, exhibits a sprawling diversity, which is only exaggerated by the historical vagaries of naming. "Manuscript editing", "page layout", "command languages" -- what underlying structure

exists in this collection of tasks, and in others that do not yet appear in the table? The vertical dimension of the table is not intended to be an unstructured list. One major theoretical concern of AIP is the development of a scientifically relevant taxonomy of tasks that would produce some order in this exploration. Such a taxonomy would tell us, for example, whether the requisite basic knowledge was on hand to apply psychology to a given office system interface or whether specific additions to that knowledge would be required.

Quick studies are crucial for an applied psychology, but not all steps toward creating the applied science can be taken that way. A few basic issues before us need substantial effort (say, of the size and depth of a Ph.D. thesis) before they are well enough understood to be fully assimilated into the theory. Consider user errors. Almost all of the AIP studies thus far have concentrated on the error-free performance time of a given task. Setting errors aside was an important step in making progress on that aspect of performance, but errors are themselves a crucial aspect of performance and must be dealt with. Since errors are relatively rare and highly diverse events, the research problem is how to collect a substantial body of error data for analysis. Once we have the data, there is the problem of building a model for predicting where errors will occur and how frequently. These problems are solvable, we believe, but require extended studies. Another basic issue is the learning of new systems. The research problems are different than for errors, but they also require a large initial push to get the issue under control. And there are other issues. Only after we get over the initial hump on these issues is it possible to proceed profitably with small studies that involve them.

Therefore, the nature of the AIP effort is *research*, including a great deal of basic research at that! But all of the basic research is focussed on providing the capability for doing relevant, quick, high-payoff application studies.

Conclusion

We have stated a vision of an applied psychology for office information systems. Such an applied psychology, come to realization, would lead to systems that humans would find easier to use and to learn. We do not know how much this would increase the productivity and reliability of the total office operation. That cannot be a conclusion of an essay that states a vision. General observation attests to the need for good interfaces. Faith in science and its relevance to application insists that the solution to ignorance is systematic and tested knowledge. But no one has seen the effects of having a really good theory of the human user and no one knows how big a part of the total office information system it can affect.

We have stated the vision as clearly as we could: an applied psychology based on theory, admitting of calculation, applicable to the full range of design issues, and part of the regular designer's kit of routine techniques. We have made clear that a "theory-based calculational psychology" means just what any engineer would expect: relevant aspects of systems can be characterized by formulas, which can then be used in flexible ways. Some examples from our work provided illustrations. We showed how to characterize the time a user takes to edit a manuscript and how that same theory can be used to estimate the time to do page-layout in a proposed system. We showed how to characterize the time to point with a continuous cursor-control device, and how the formula indicated what aspects were already optimal.

The vision is clear enough so we could describe examples of possible applications. We described a Simulated User, a computer program that embodies much of what is known about how humans use office systems; it can be used in connection with standard design simulations to make more accurate design explorations and evaluations. We described a general notational system for command languages that would permit the characterization of the User's Model of the office system. Given this, design of new systems could proceed by first laying out the User's Model -- a truly new design capability. We described how the knowledge about the psychology of the user could be cast in a form to be used by the practicing designer, rather than just by the professional psychologist.

Given this vision, we sketched by means of a table what AIP has been doing in the three years of its existence. This table laid out the ingredients needed to support application in any task area. We explained a fundamental strategy of AIP, which is to iterate quickly over a series of attempts to extend our knowledge and theory to new domains. We showed that the basic nature of the AIP effort is research, but research that provides the capability for effective application.

AIP is now three years down the road in creating an applied psychology. We have a vision of what that applied psychology will be like. We think that it is an attainable goal, that it is useful -- even crucial -- for Xerox, that we have made considerable progress toward that goal in the three years, and that the road to it leads straight ahead.

Reading Further

For the interested reader, the following are some basic references into the cognitive psychology literature:

P. N. Lindsay & D. A. Norman.
Human Information Processing.
Academic Press, 1972.

A. Newell & H. A. Simon.
Human Problem Solving.
Prentice-Hall, 1972.

R. B. Klatsky.
Human Memory.
W. H. Freeman, 1975.

A. T. Welford.
Fundamentals of Skill.
Mentheun (London), 1968.

Many of the topics discussed in this essay are based on work we have done over the last three years. This work is documented in an AIP memo series. The AIP memo numbers for specific studies are shown on the table on the next page. Below are listed the memos most likely to be of interest to readers of this essay:

A. Newell.
Notes on a proposal for a psychology research unit.
AIP Memo 1, January 1971, 11 pages.

S. K. Card, T. P. Moran, & A. Newell.
The manuscript editing task: a routine cognitive skill.
SSL Report 76-8 (AIP Memo 82), December 1976, 80 pages.

S. K. Card, T. P. Moran, & A. Newell.
Zeroth-order analysis of a document composition task.
AIP Memo 83, August 1976, 31 pages.

S. K. Card, W. K. English, & B. Burr.
*Evaluation of mouse, joystick, step keys, and text keys
for use in text selection on a CRT.*
SSL Report 77-1 (AIP Memo 95), April 1977, 27 pages.

AIP STUDIES

(June 1977)

BASIC INGREDIENTS

APPLICATIONS

X = completed
c = current
p = planned

TASK AREAS AND STUDIES	Exploration					Calculation			Memos*
	Theory		Verification			Evaluation			
			Data Base			Design Use			
			Tools						

Manuscript Editing									
Early Poet studies	X	X	34
Rand editor prediction	X	.	.	53,56
Benchmark studies	.	.	.	X	.	.	X	.	54,79,93
Woodstock evaluation (Kimball)	X	.	**
Data handling programs	X	.	.	.	--
Poet method selection	X	X	X	X	67
Poet modelling study	.	X	X	X	82
Bravo individual differences	X	.	.	X	86
Bravo model	c	c	c	c	88,97
Page layout prediction	X	X	.	.	83
Cursor device study	X	X	X	X	.	X	X	X	92,95
Poet learning studies	X	X	37,94,98
Feedback dictation study	X	78
Command Language Systems									
Design evaluation workshop	X	X	.	.	--
SDD user interface group	X	X	***
Critical incident study	c	p	.	--
Command Language Grammar	.	c	p	p	--
Graphics & Displays									
Line drawing task study	X	.	.	X	85
Display studies	p	p	p	p	.	p	p	p	--
General									
System evaluation workshop	X	--
L* Alto implementation	X	.	.	.	--
Simulated user exploration	X	90,97
Parallel behavior study	c	c	c	--
Cognitive handbook	p	p	.	p	.	p	p	p	--
Task taxonomy study	p	p	.	.	.	p	.	.	--
Errors study	p	p	p	p	--

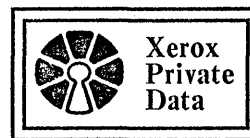
*AIP Memo numbers

**SSL Memo

***SDD Memo

ALTO SUBSYSTEMS CATALOG

January, 1978



Filed on [MAXC]<AltoDocs>SubsystemsCatalog.press

This catalog lists and briefly describes the various Alto subsystems. A subsystem is defined to be any program that runs on the Alto whether under control of the standard executive or as a standalone. Each subsystem has an entry of the following format:

PROGRAM NAME: DESCRIPTION
DOCUMENTATION

To simplify locating a piece of software to perform a specific function, a functional cross reference of Alto subsystems is provided beginning on page 8.

New subsystems are continually being developed throughout the Whole Alto World. This catalog is maintained and distributed by the Whole Alto World coordinator (Frank Ludolph). If you have or know of a subsystem which you feel should be included, please provide the above information to the coordinator. The following criteria should be met before a subsystem is cataloged and distributed:

1. The program should be in general use at the installation.
2. Support should be provided to at least fix major bugs.
3. Satisfactory documentation must be available and up to date.
4. Management must indicate the sensitivity of the item to outside disclosure.

The subsystems listed are generally available from your local IVY station or [MAXC] under the <ALTO> directory; the supporting documentation is found in the <ALTODOCS> directory unless otherwise noted. If you do not have access to an IVY station or MAXC, contact the coordinator for subsystems/documents requests.

Italicized program names indicate entries that are new or significantly altered.

AIS: a driver subsystem which interacts with the user to perform a set of standard operations on imaginal data stored as AIS (Array of Intensity Samples) files.
DOCUMENTATION: [MAXC]<AIS>AIS-Manual.ears.

ANALYZE: a part of the Design Automation System that transforms logic diagrams produced using SIL into a file which can be input to the GOBBLE wirelister.
DOCUMENTATION: [MAXC]<SIL>SilManual.press.

ANSRV: a program that enables the Alto to use a Nova as a "remote batch" computer, i.e. the Alto can ship jobs to the Nova for execution and retrieve the results following job completion.
DOCUMENTATION: ANSRV.tty or Subsystems.ears.

APROM: Superseded by PROM.

ASM: an assembler for the Alto machine language which produces relocatable files compatible with the BCPL loader, BLDR.
DOCUMENTATION: ASM.tty or Subsystems.ears.

BCPL: a compiler for Alto BCPL language which produces files relocatable with the BLDR loader.



DOCUMENTATION: BCPL.tty/.ears.

BLDR: a loader for the relocatable files produced by BCPL and ASM.

DOCUMENTATION: BCPL.tty/.ears.

BRAVO: a text editor having extensive formatting and hardcopy facilities.

DOCUMENTATION: ALTO User's Handbook, BRAVO Course Outline, BRAVO.ears and BRAVOSUMMARY.ears.

BTREETEST: a B-Tree dictionary maintenance program used to support the PROOFREADER data base.

DOCUMENTATION: ProofReader.tty.

BUILD: a part of the Design Automation System that helps with the data management aspects of building boards and keeping the design automation data files current.

DOCUMENTATION: [MAXC]<SIL>SilManual.press.

BUILDBOOT: a program for constructing type B bootfiles from either an executable (BLDR out) file or a segment file.

DOCUMENTATION: BUILDBOOT.tty or Subsystems.ears.

CALCULATOR: a bootfile that pictures a TI SR-52 on the display which is operated by using the mouse to select the appropriate keys. It is not programmable.

DOCUMENTATION: SR-52 Manual.

CHAT: a program for establishing PUP Telnet connections between a pair of cooperating parties. Its chief function is to permit Alto users to talk to MAXC.

DOCUMENTATION: ALTO User's Handbook, CHAT.tty or Subsystems.ears.

CLEANDIR: a program to garbage collect a disk file directory (but not disk space).

DOCUMENTATION: Subsystems.ears.

CLEANLOG: a program to compact Sys.Log (or others of identical format), it is essentially obsolete by the release of Operating System 14.

DOCUMENTATION: CLEANLOG.tty or Subsystems.ears.

COPYDISK: a program for copying entire diskpacks. It will copy from one drive to another on the same machine, or between drives on separate machines via a network using Diablo Model 31/44 and Trident T-80/T-300 disks.

DOCUMENTATION: COPYDISK.tty or Subsystems.ears.

COPYFROMDRIVE1: a program to copy an individual file from DP1 to DP0 of a dual disk Alto.

DOCUMENTATION: Subsystems.ears.

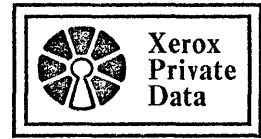
CREATEFILE: a program to create a file of a given size, attempting to allocate it on consecutive disk pages.

DOCUMENTATION: Subsystems.ears.

CRTTEST: a diagnostic program used to adjust the Alto display linearity. Three different sized grids are displayed in rotation (press any key).

DOCUMENTATION: none.

CRUMPLE: a program to compress and, optionally, encrypt data files. The resulting files can be stored or transmitted but must be expanded and decrypted before processing by Alto programs.



DOCUMENTATION: Crumple.press.

DDS: a program to manage an Alto diskpack. Facilities are provided to display filenames, lengths, creation-read-write dates, and contents, internal operations such as delete, and rename, and external operations such as Send and Execute.

DOCUMENTATION: ALTO User's Handbook or DDS.tty or Subsystems.ears.

DMT.BOOT: a memory diagnostic and statistics gathering program.

DOCUMENTATION: DMT.tty or Subsystems.ears.

DPRINT: a program to type text files on the Diablo Hytype printer.

DOCUMENTATION: DPRINT.tty or Subsystems.ears.

DRAW: an interactive illustrator program for creating black-and-white or color pictures composed of lines, curves, and text captions. The illustrations can be output to a one page press file.

DOCUMENTATION: ALTO User's Handbook plus DRAWnews.ears, an on-line manual (part of the DRAW package), and DRAW-Summary.ears.

EDP: an Ethernet interface diagnostic.

DOCUMENTATION: None.

EMPRESS: a program to send press and text, e.g. bravo format, files to a press printing server. Simple formatting options such as Tab and FormFeed are available.

DOCUMENTATION: EmPress.tty or Subsystems.ears.

EXECUTIVE: the Alto command processing subsystem, the intermediary by which users generally invoke other subsystems and perform several operations on the Alto file system. Normally invoked by the boot operation.

DOCUMENTATION: Executive.tty or Subsystems.ears.

FIND: a subsystem to search one or more text files for a user supplied string at very high speed and then display each line containing an occurrence of the pattern on request.

DOCUMENTATION: FIND.tty or Subsystems.ears.

FRED: a part of the Font Creation System, it is used to create and/or edit "splines" (i.e. outlines) of characters.

DOCUMENTATION: [MAXC]<GR-DOCS>Fred.ears.

FTP: a file transfer program to store and retrieve files between an Alto and another Alto, Maxc, or IFS station. It also supports a Telnet connection that is similar to CHAT in purpose and operation.

DOCUMENTATION: FTP.tty or Subsystems.ears.

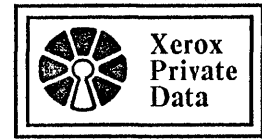
GOBBLE: a part of the Design Automation System that generates a wirelist and routing information for a single board given one or more node list files generated by ANALYZE.

DOCUMENTATION: [MAXC]<SIL>SilManual.press.

GEARS: an program to compose text files and transmit them to the EARS printer via the Ethernet.

DOCUMENTATION: Gears.tty or Subsystems.ears.

GYPSY: a modeless text editor using both keyset and mouse, and having a "filing cabinet" interface which provides some file management facilities beyond the normal Alto filing system. Used in applications where limited formatting facilities are required such as programming.



DOCUMENTATION: Under development.

ICARUS2: a part of the ICARUS2 System, it is an interactive program for actually laying out printed circuits and manipulating the resulting files.

DOCUMENTATION: [MAXC]<ICARUS>Icarus2doc.press, ICtools.press.

IFD2: a part of the ICARUS2 System that turns an ICARUS2 file into human readable form, describing each symbol and its contents.

DOCUMENTATION: [MAXC]<ICARUS>IFD2doc.press.

IFS: the IVY File System server that provides one end of the file transfer facility and maintains the files and directories on Trident T-80/T-300 disks.

DOCUMENTATION: [MAXC]<IFS>IFSdocuments.press.

IFSSCAVENGER: a subsystem to check and correct Trident T-80/T-300 diskpacks from the IVY and Trident File Systems.

DOCUMENTATION: [MAXC]<IFS>ScavOp.press.

INSTALLSWAT: an installation program to install the SWAT debugging system on you disk.

DOCUMENTATION: None.

KEYTEST: a diagnostic program that displays the Alto I keyboard, keyset and mouse. The depressing of any key(s) is reflected by inverting (white to black) the display of that key on the screen.

DOCUMENTATION: none.

LISTSYMS: a programming aid to convert a .Syms file (produced by BLDR) to a useful, human readable form.

DOCUMENTATION: ListSyms.tty or Subsystems.ears.

LOGICPROM: Superseded by PROM.

MADTEST: a bootfile diagnostic that runs tests on an Alto's RAM, ALU, and emulator.

DOCUMENTATION: Enter "?" while running.

MAILCHECK: a simple subsystem that checks for mail at some other host (e.g. Maxc) via the Ethernet.

DOCUMENTATION: MailCheck.tty or Subsystems.ears.

MARKUP: an illustrator used to add pictures consisting of lines, areas, mouse tracks and text captions to formatted documents, i.e. Press files. It may also be used to simply display press files.

DOCUMENTATION: Alto User's Handbook.

MIKE: a part of the ICARUS2 System that transforms ICARUS2 files into a form suitable for the Mann 3000 pattern generator.

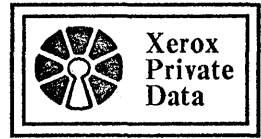
DOCUMENTATION: [MAXC]<ICARUS>MikeUserDoc.press, MikeDoc.press.

MOVETOKEYS: Obsolete, non-functional.

MU: the Alto Microcode assembler.

DOCUMENTATION: MU.tty or Subsystems.ears.

NETWORK EXECUTIVE: the executive obtained by booting from the Gateway over the Ethernet that provides a convenient way to call "bootfiles" such as FTP or COPYDISK.



DOCUMENTATION: NetExec.tty.

NPGR: a part of the SIL system that sends SIL files to the Ears printer.
DOCUMENTATION: [MAXC]<SIL>SilManual.press.

NPPR: a part of the Design Automation System that generates a Press file from a SIL file for printing on a press printer.
DOCUMENTATION: [MAXC]<SIL>SilManual.press.

OEDIT: a subsystem for displaying and modifying Alto files in octal. Up to four files may be simultaneously viewed while one of them may be modified.
DOCUMENTATION: Oedit.tty or Subsystems.ears.

ORBITTEST: the ORBIT interface diagnostic.
DOCUMENTATION: [IFS]<SPRUCE>ORBITtest.press.

PACKMU: a program convert the output of MU (an MB file) to a "packer RAM image" which is easy to load into the RAM using RPRAM.
DOCUMENTATION: PackMU.tty or Subsystems.ears.

PEEK: a program which listens to the Ethernet for PeekReports and EventReports. It can also serve as a bootserver and Ethernet Echo server for use with EDP.
DOCUMENTATION: DMT.tty.

PEEKUP: a small subsystem enabling one to peek at Pups going to and from a particular Ethernet host; a debugging aid for new Pup software.
DOCUMENTATION: PeekPup.tty or Subsystems.ears.

PEEKSUM: a subsystem that summarizes the error reports sent to PEEK by DMT.
DOCUMENTATION: DMT.tty.

PREPRESS: a part of the Font Creation System that takes "spline character definitions", usually created by FRED, and generates scan-converted characters, spline and character dictionaries, readable listings describing the dictionary's content, and a "widths" file for use by text formatting programs.
DOCUMENTATION: [MAXC]<GR-DOCS>PrePress.ears.

PRESS: a subsystem to print full press files on press printers.
DOCUMENTATION: [MAXC]<GR-DOCS>PressOps.ears.

PRESEEDIT: a program to combine Press files together, convert Ears files (generated by Pub and Bravo) to Press format, selecting certain pages from a Press or Ears file, or add extra fonts. The output is a Press file.
DOCUMENTATION: PressEdit.tty or Subsystems.ears.

PROM: a subsystem to edit microcode, drive the Alto PROM blower and verify PROMs.
DOCUMENTATION: PROM.bravo.

PROMDIAG: Superceded by PROM.

PROOFREADER: an interim English text proofreader that produces an output file listing the questionably-spelled words.
DOCUMENTATION: ProofReader.tty.



PUPTEST: a PUP protocol and network integrity test program.
DOCUMENTATION: None.

PUT: a program for transferring files between the disks of a dual-drive Alto.
DOCUMENTATION: Internal to the program.

QED: an in-core line editor used primarily for programming. The file is limited to about 1500 lines of BCPL.
DOCUMENTATION: QED.tty or Subsystems.ears.

RAMLOAD: a microcode loader that uses the output of the microcode assembler, MU.
DOCUMENTATION: RamLoad.tty or Subsystems.ears.

RAMTIMING: a bootfile diagnostic program to test the Alto RAM. Its function is also performed by the more comprehensive MADTEST.
DOCUMENTATION: None.

READPRESS: reads Press files and displays a text-listing of the entity commands, DL strings, etc.
DOCUMENTATION: Subsystems.ears.

RENAME: a functional replacement for the EXECUTIVE supplied "rename.~" used to quickly rename files on the Alto disk.
DOCUMENTATION: Subsystems.tty.

RPRAM: a microcode loader that loads a packed RAM image (generated by PACKMU) into the RAM after checking the constant memory.
DOCUMENTATION: PackMU.tty or Subsystems.tty.

SCAVENGER: a subsystem for checking and correcting Alto disk packs.
DOCUMENTATION: Scavenger.tty or Subsystems.ears.

SEARS: a part of the Ears printing system run on the Alto, Palo, to start the Ears system.
DOCUMENTATION: Gears.tty or Subsystems.ears.

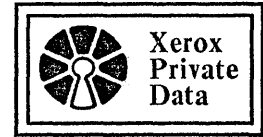
SETTIME: this simple subsystem attempts to obtain the date and time from some other host on the Ethernet.
DOCUMENTATION: SetTime.tty or Subsystems.ears.

SIGMA: a subsystem to transfer arbitrary files between an Alto and a SIGMA 3 over the Ethernet.
DOCUMENTATION: "Ethernet Software for Data Transfer between the SIGMA 3 and an ALTO", a Xerox Internal Report, Accession No. X7704459.

SIL: a part of the Design Automation System, it is an illustrator for the creation of logic and line diagrams. The output may be processed by NPPR to generate Press files or processed by ANALYZE for circuit design.
DOCUMENTATION: [MAXC]<SIL>SilManual.press.

SORT: a very small subsystem which will sort files containing less than 1000 entries delimited by a carriage return.
DOCUMENTATION: Subsystems.ears.

SPRUCE: a printer server that utilizes the ORBIT buffer to drive Press printers, e.g. Dover and Sequoia.



DOCUMENTATION: Under revision.

SWAT: a emulator-level code debugger with BCPL oriented features used with the Alto operating system.

DOCUMENTATION: Swat.tty or Subsystems.ears.

SYS.BOOT: the operating system boot file on the Alto disk.

DOCUMENTATION: None.

TFU: a file utility used to initialize a Trident pack with a virgin file system and to perform various file copying, deleting, directory listing operations. This is not a part of the IVY System, rather it initializes and maintains packs operated on by the TFS package.

DOCUMENTATION: TFS.tty or Sybsystems.ears.

TRANSFILE: a part of the ICARUS2 System that translates the intermediate files generated by Mike into human-readable form. The files it produces are fully instantiated.

DOCUMENTATION: [MAXC]<ICARUS>TransfileDoc.press.

TRIEX: a Trident diagnostic used to initialize, verify, and exercise Trident disk packs and drives.

DOCUMENTATION: Self-contained.

TYPE: a functional replacement to the Executive supplied "type.~" that displays a larger page, suppresses Bravo trailer information, can skip forward and backward, etc.

DOCUMENTATION: TYPE.tty.

UGH: an in-core text editor utilizing both mouse and keyset. While some formatting facilities are available it is used primarily for programming.

DOCUMENTATION: UGH.tty.

VIEWDATA: a subsystem to display on the Alto screen three-dimensional data stored as a two-dimensional array of single-word values.

DOCUMENTATION: ViewData.tty.

VIEWIC: a part of the ICARUS2 System that displays the data created by Mike on the Alto screen, simulating the actions of the pattern generator.

DOCUMENTATION: [MAXC]<ICARUS>ViewicDoc.press.

VPRINT: a subsystem to output text files such as .TTY, UGH, BRAVO, or GYPSY, to a Versatec printer.

DOCUMENTATION: Under development.



The following list of Alto II subsystems is organized according to the general function they perform. Because many subsystems perform more than one function or a function may be thought of in a variety of ways, an item may be listed more than once.

The major functional headings are:

Document Creation	Hardware Design	Messages
Files	Hardware Diagnostics	Printing
Font Creation	Hardware Drivers	Programming
		Recovering

DOCUMENT CREATION

EDITORS

TEXT

BRAVO - Rich in formatting features.
 GYPSY - Features to handle groups of files (e.g. chapters or modules).
 PROOFREADER - Produces an output file of questionably spelled words.
 QED - A line editor.
 UGH - An in-core editor that uses the keyset for command input.

GRAPHIC

DRAW - Pictures composed of lines, curves, text and smoothed mouse tracks.
 FRED - A Spline editor for font work.
 MARKUP - Dot pictures of lines, areas, text, and mouse tracks.
 SIL - For creating diagrams composed of lines with text captions.

IMAGINAL

AIS - Image manipulation, printing, Press file creation.

PAGE MAKEUP

MARKUP - Create new or move pre-existing diagrams along side existing text.

MERGING

AIS - Merges bitmap files (e.g PRESS output and AIS files).
 EMPRESS - Append press files to personalized coversheets.
 PRESSEEDIT - Generate Press file from pages of other Press files.

PRINTING - see PRINTING below.

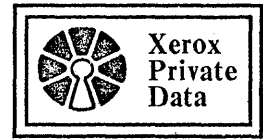
FILES

DISPLAY

OEDIT - Alto files in octal.
 MARKUP - Press files.
 PRESS - Press files.
 TYPE - Text files.
 VIEWDATA - Three-dimensional information stored as a matrix of values.

TRANSFER

COPYDISK - Copies whole disks, Diablo Models 31/44 and Trident T-80/T-300.
 FTP - Copies a file between Altos, Alto-IVY, and Alto-MAXC.
 GYPSY - Transmits its files to a Communicating 800 ETS.
 PUT - Copies files between disks of a dual drive Alto.
 SIGMA - Copies files between Alto and SIGMA 3.
 TFU - Copies files between Trident drives.

**ALTO FILE SYSTEM**

CLEANDIR - Garbage collect disk directory.
 CREATEFILE - Adds new file of specified size on consecutive pages, if possible.
 CRUMPLE - Compresses and optionally encrypts a file.
 EXECUTIVE - Delete files, list directory.
 FIND - Locates and displays lines containing specified text string.
 FTP - Transfers files between Alto and Alto, IVY, or MAXC.
 OEDIT - Display and modify file in octal.
 PUT - Deletes, renames, and copies files between disks of a dual drive Alto.
 RENAME - Quickly renames a file.
 SCAVENGER - Checks and corrects the disk.
 SIGMA - Transfers files between Alto and SIGMA 3.
 SORT - Sorts up to 1000 items delimited by carriage returns.
 TYPE - Displays contents of text files.

IVY FILE SYSTEM

IFS - The server.
 IFSSCAVENGER - Checks and corrects Trident T-80/T-300 disks.
 TFU - Initialize directory and verify disk.

TRIDENT FILE SYSTEM

IFSSCAVENGER - Checks and corrects Trident T-80/T-300 disks.
 TFU - Initialize and list directory, verify disk, copy and delete files.

RECOVERY

IFSSCAVENGER - Check and correct disks of IVY and Trident file systems.
 SCAVENGER - Check and correct disk of Alto file system.

FONT CREATION**SPLINES**

DRAW - Create splines using mouse or knots.
 FRED - Create and edit splines, create font files.

BITMAPS

PREPRESS - Scales and rotates splines, converts to and edits bitmaps.

DEVICE FORMATS

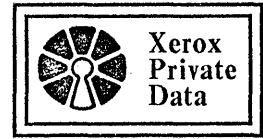
PREPRESS - Creates printer and display fonts from bitmaps.

HARDWARE DESIGN**CIRCUIT BOARD**

SIL - Create and edit logic diagrams.
 ANALYZE - Converts SIL drawing for GOBBLE, generates SIL of unassigned pins.
 GOBBLE - Generates wirelist and routing information.
 BUILD - Aids data management aspects, keeping data files current.
 NPGR - Sends a SIL file to Ears.
 NPPR - Generates a Press file from a SIL file.

INTEGRATED CIRCUIT

ICARUS2 - Layout integrated circuits.
 IFD2 - Generates human readable form of ICARUS2 files.
 MIKE - Transforms ICARUS2 file to form for Mann 3000 pattern generator.
 TRANSFILE - Generates human readable form of MIKE file.
 VIEWIC - Displays MIKE output.
 ICGERB - Generates Gerber photoplotter output from ICARUS files.

HARDWARE DIAGNOSTICSUSER

CRTTRST - Displays a rectangular grid. (.boot/.run files).
 DMT - Memory diagnostic that transmits results to PEEK. (Alto/server boot files).
 EDP - Ethernet interface diagnostic.
 KEYTEST - Keyboard diagnostic. (.boot/.run files).
 MADTEST - Diagnostic for RAM, ALU, and emulator. (.boot/.run files).
 TRIEX - A Trident T-80/T-300 diagnostic.

INSTALLATION

DISKTEST - Bootfile diagnostic for the Diablo Model 31.
 ORBITTEST - An ORBIT Interface diagnostic.
 PEEK - Collects PeekReport/EventReport packets on <filename>.
 PEEKSUM - Summarizes DMT error reports collected by PEEK.
 PUPTEST - Ascertain status of network servers.

HARDWARE DRIVERS - See the Alto Hardware Catalog under appropriate device.

MESSAGES (Requires MAXC account)SENDING

CHAT - Accesses SNDMSG on MAXC.

RECEIVING

MAILCHECK - Interrogates MAXC for new mail.
 CHAT - Accesses MSG on MAXC to retrieve mail.

PRINTINGREMOTEREFORMATING

PREPRESS - Generates a Press file from .tty and many .ears files.

EARS (Palo Alto only)

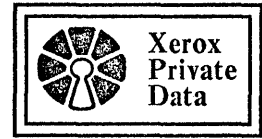
BRAVO - Sends the workfile to Ears.
 GEARS - Format and send text file to Ears.
 PRINT - Send Press file to Ears.
 SEARS - Ears server.

PRESS

BRAVO - Sends the workfile to SPRUCE.
 EMPRESS - Sends the specified Press and/or Text files to SPRUCE.
 GYPSY - Sends the workfile to SPRUCE.
 SPRUCE - A Press server for Dover/Sequoia/Pimlico.

LOCAL

BRAVO - Prints the workfile on the attached Diablo HyType.
 DPRINT - Prints text files on the attached Diablo HyType.
 GYPSY - Prints the workfile on the attached Diablo Hytype.
 PRESS - Prints Press files on slot and Versatec printers.
 VPRINT - Prints text files on Versatec printers.

**PROGRAMMING**

EDITORS - See DOCUMENT-EDITORS-TEXT above.

ASM/BCPL

ASM - Alto machine language assembler.
BCPL - BCPL compiler.
BLDR - Loader for ASM and BCPL relocatable files.
BUILDBOOT - Generates a type B bootfile.
SWAT - an emulator level, BCPL oriented debugger.
INSTALLSWAT - Installs SWAT on a disk.
LISTSYMS - Converts .SYMS files to human readable form.

MU

MU - Microcode assembler.
RAMLOAD - Loads RAM with MU output.
PACKMU - Converts MU output for RPRAM.
RPRAM - Loads RAM with RPRAM output.

DEBUGGING AIDS

SWAT - An emulator level, BCPL oriented debugger.
LISTSYMS - Converts .SYMS files to human readable form.
BTREETEST - A B-tree dictionary maintenance program.
PEEKPUP - Peeks at PUPs going to and from a specific Ethernet address.
PUPTEST - Interacts with new subsystems that use the PUP protocol.
READPRESS - Displays entities within a Press file.

RECOVERING

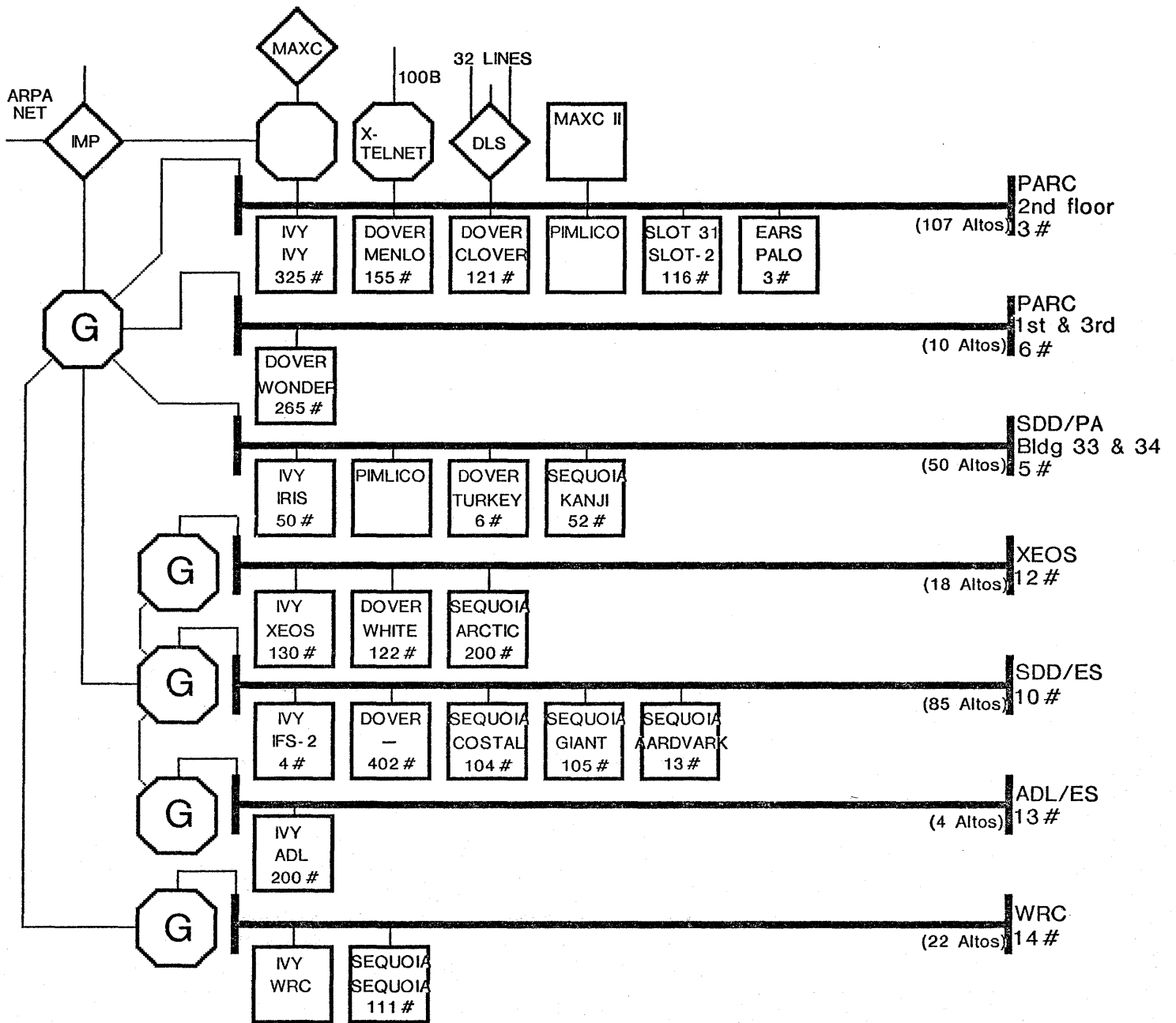
DISK FAILURES - See FILES-RECOVERY above.

SUBSYSTEMS FAILURES


BRAVOBUG - Recovers BRAVO files to point of failure.


XEROX ALTO NETWORK


February, 1978



Filed on [MAXC] <AltoDocs> AltoNetwork.press

 ALTO Server
(ALTO + Device)

 NOVA
(G is w/ pup Gateway software)

 Other Resource

Whole ALTO World Newsletter

Technology and Tools

XEROX

March 31, 1978

GENERAL NOTES

MAINTENANCE NOTES - A new subsection is being added to the Newsletter to circulate knowledge of specific hardware problems, solutions, and maintenance techniques. It's located between the **HARDWARE** and **SOFTWARE** subsections under **TOOLS**. While many of the items will be directed to the people that maintain the hardware, some will be of general interest to Alto users, so don't just skip over this section.

MESA - The information on MESA in the last Newsletter needs some clarification. Although MESA as a language is still evolving and should not be used outside SDD for tightly scheduled projects, it is a reasonably robust and complete system and will be highly compatible from the Alto to successor machines. As such it should be seriously considered for both short and long term projects. Remember, the WAW coordinator, Frank Ludolph, is the MESA contact for non-SDD, non-PARC users.

TOOLS

HARDWARE

SEQUOIA BUILD - This is the last opportunity to indicate your interest in purchasing a Sequoia from the proposed October build. Contact Sam Losh at XEOS now, Intelnet 8*844-2501.

HUSHING THE TRIDENT - Jensen Engineering will custom build quiet boxes for the T80 and T300 drives. Externally the units will look identical. The prototype will be "Alto" grey but the production color(s) is not yet certain. Preliminary price estimates are \$450 to \$500 for the T80 box and \$350 to \$400 for the T300 box. (The T80 box costs more because a platform is necessary to raise it to the same height as the T300.) If interested please send a message to Barbara <BAIRD>; quantity orders will lower the price. *Don't wait for an "iffy" second build.*

BUILDING REGULATIONS AND THE ETHERNET - The 1978 National Fire Code states that electrical wiring within the environmental airspace must be within conduit. In a building with a drop ceiling, the space above the ceiling is part of the environmental air space if that space is used for air return by the airconditioning system. If both intake and exhaust ducting is used, the space above the ceiling is not considered a part of the environmental airspace. Specifically exempted is low-voltage communication cable provided the cable is approved for the purpose, i.e. it does not contribute to combustion. To be approved the cable must meet both smoke and flammability standards. Common polyethylene insulated coaxial cable is not acceptable but cable having a non-contaminating vinyl sheath has been approved.

MAINTENANCE NOTES

DIABLO 31 WRITE-HEAD CURRENT - Because Diablo intended the Model 31 to be run at a slightly lower recording density than used in the Alto, the write-head current beyond track 128 may be insufficient for reliable operation. Diablo advises that the write-head current can be raised by altering the value of resistor H-64 on the J10 board, starting with 1K Ω and reducing if necessary. Resistor F-63, part of the same voltage dividing network, should not be cut as this drives the associated transistors to full on, altering their characteristics and causing splatter (the reason for the current cut in the first place).

HARDWARE CHECKOUT BY THE USER - There are several diagnostic programs that users can run to verify that the pieces of their Alto are running properly: DMT, CRTTEST, KEYTEST, and MADTEST. These are available from boot servers such as Gateways. To execute them, boot over the ethernet (boot while depressing the BS and quote keys) or type the 'netexec' command to the Executive. At this point the NetExec will appear on the screen. Type the name of the diagnostic and you're off and running. (Enter a '?' to list the boot files that can be called by the NetExec; the diagnostics listed above should appear. *Do not use DISKTEST. It's intended only for maintainers, requires documentation to use, and could overwrite a readied disk.*) CRTTEST and KEYTEST are used to checkout the workstation and are very simple to use (see below). DMT and MADTEST checkout the Alto itself and will be described next month.

CRTTEST draws parallel vertical and horizontal lines. The thing to look at is the sharpness of the lines (are they fuzzy?) and the shape of the boxes circumscribed by the lines. The boxes should be square, not tall or wide or diamond shaped (romboid). There will be a little distortion at the corners so don't worry about that. Depress the space bar (or any other key for that matter) and the lines will be redrawn with a different spacing; there are three different spacings. To quit, boot.

KEYTEST tests the action of the keyboard, keyset, and mouse. When the program begins, these items will be drawn on the screen (you may have to move the mouse to find it's diagram). It was recently enhanced to display either the Alto I or Alto II keyboard as appropriate. Depress each key, one at a time; the corresponding key on the display should turn black. If it stays white, or more than one key turns black, there is a problem. The latter is particularly a problem with the Alto II keyboard. To quit, boot.

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local IVY server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [XEOS].

NEW RELEASE: AISshow.run - This new addition to the AIS (Array of Intensity Samples) system was written by Paul Roetling. As the name implies, the program displays AIS picture files on the Alto screen. If the image is larger than the screen, the image will be demagnified using a nearest neighbor algorithm. It operates on either 1 bit/pixel and 8 bit/pixel images; the 8 bit/pixel images will be displayed using a Floyd halftone routine. Demagnified 1 bit/pixel images may show substantial Moire patterns. Retrieve [WRC]<AIS>Subsystems>AISshow.run. The documentation, <AIS>Memos>AISshow.memo is appended to the Newsletter.

ReReleases - Subsystems

AISdump - An error which printed some incorrect values for windows of odd length was fixed. The new version, 1.1, is available from [WRC]<AIS>Subsystems>AISdump.run.

COPYDISK - The "[thisHost]device" bug has been corrected and the default value of WRITEPROTECT is now TRUE. This subsystem is available from boot servers.

DDS - This new version conforms to the new time standard. Retrieve <Alto>DDS.run.

DMT - This new version conforms to the new time standard. Retrieve <Alto>DMT.run.

EXECUTIVE - This new version conforms to the new time standard, fixes a few bugs, and has some enhancements including the ability to load CHAT, FTP, Scavenger, and NetExec over the ethernet, a FILESTAT command to report file attributes, and a SETTIME that obtains the time over the ethernet (delete SetTime.run). Use <Alto>NewOS.cm to update this subsystem as well as the new OS/15 and FTP. The documentation, <AltoDocs>Executive.tty, has been revised.

FTP - The 21 March 78 version fixes more file date bugs and contains new features including a typescript of the user window, new commands (Open, Close, and Compare) in the command line, and improved command line error handling. This subsystem will be updated when installing the new operating system. See the revised documentation, <AltoDocs>FTP.tty.

KEYTEST - This diagnostic, available from boot servers, has been enhanced to display the correct keyboard, i.e. Alto I or Alto II. Retrieve <Alto>KeyTest.run only if boot servers, e.g. gateways, are not on your ethernet.

LISTSYMS - This new version conforms to the new time standard. Retrieve <Alto>ListSyms.run. The documentation, <AltoDocs>ListSyms.run has been revised.

MICRO/MICROD - A primary reason for the rerelease is to alleviate a space constraint which prevented certain diagnostics from assembling. Also the filename extension, .MC, is now defaulted rather than forced. Retrieve <Alto>MICRO.run. The changes document, <AltoDocs>MICRO.tty has been updated.

MU - A bug which failed to give an error message when a semicolon was left off the end of a predefinition was corrected, the filename extension now defaults to .Mu, and the listing file contains constants sorted by value as well as by address. Retrieve <Alto>MU.run and the revised documentation, <AltoDocs>MU.tty.

NETEXEC - This boot file has been enhanced to poll boot servers for the boot files they can supply, and has some new user command functions such as PROBE and HOST. No .run file need be retrieved. The new documentation is available on <AltoDocs>NetExec.tty.

OPERATING SYSTEM - This version fixes bugs in the file date code which caused the FTP update command to fail. Retrieve and execute <Alto>NewOS.cm. Verify that there are at least 300 free pages on your disk before executing the command file. This will also update FTP and Executive.

PEEK - This new version conforms to the new time standard. Retrieve <Alto>PEEK.run.

Whole ALTO World Newsletter

PRESSEEDIT - The experimental version reported last month has been officially released. It enables the merging of one page graphics onto a text document at any location on the document page, so now it's an easy matter to edit text after inserting graphics (re-edit the Bravo file, make a Press version, and remerge). It also fixes bugs concerning Private Data Labels and very complex pages. Retrieve <Alto>PressEdit.run and the new documentation, <AltoDocs>PressEdit.tty.

PROM - The nature of the changes are unknown to me. Retrieve <Alto>PROM.run.

READPRESS - The new version properly recognizes Press file elements that have just come into use. Retrieve <Alto>ReadPress.run.

TRIEX - This release incorporates the new time standard. Do not use it under OSs prior to version 14. Retrieve <Alto>Triex.run and updated documentation, <AltoDocs>TFS.tty.

TFU - This release incorporates the new time standard. Do not use it under OSs prior to version 14. Retrieve <Alto>TFU.run and updated documentation, <AltoDocs>TFS.tty.

ReReleases - Packages

PUPPACKAGE - The new version contains changes to the BSP code that improve performance when communicating through Gateways and also fixes several bugs. Several of the modules have been broken into smaller pieces to permit substantial overlaying. The documentation, <AltoDocs>PupPackage.tty, has been updated.

TIME - The new version conforms to the new time standard. If you are responsible for subsystems that deal with time in any way, you are requested to begin converting such subsystems to use the new software. The backward compatibility measures that have been implemented in OS 14 will cease to work correctly on April 30, 1978, so it is desirable that the revised subsystems be released well before that time. Announcements of new releases should include a warning that they will not work under pre-OS 14 versions. Load <Alto>Time.dm. The packages is documented in <AltoDocs>Time.tty; the new time standard is described in <AltoDocs>AltoTime.Bravo.

TFS - This release contains the modifications necessary to conform to the new Alto time standard. It will run only under OS 14/15. Also, the microcode source file has been broken up to facilitate combining it with other microcode, however the microcode binary is unchanged. Load <Alto>TFS.dm. Updated documentation is available on <AltoDocs>TFS.tty.

TECHNOLOGY

Most users are familiar with the printing of text and graphic information but there is a third type, imaginal. Continuous tone pictures (e.g. photographs) are of this third type. The problem of printing continuous tone images is essentially that of representing many shades of grey with only black ink on white paper. A solution of the printing industry was the halftone, a pattern (usually regular) of black dots which vary in size and, possibly, shape. Pictures in the newspaper are an example. Two papers are included, one which describes the halftone process and a second that presents the resolving limits of the eye, why we see groups of black dots as multi-grey tone images.

The first, *PRESS, Halftones, and You* by Joe Maleson, is something of a tutorial on halftones and the halftone process as implemented in Press. The Press specific information applies to Press 1. Press 2 has a different halftone screen pattern and no longer uses error distribution. The new screen pattern, similar to that used in the printing industry, has dramatically improved image quality. Joe has written a memo describing the halftone screen being used in Press 2. Since the electronic form does not contain the image and graphic information, contact Frank Ludolph by message, <Ludolph>, or phone, 8*923-4356, for a copy.

There is an interesting problem in the printing of papers that demonstrate methods of image representation, namely that of accurately representing the image. It is the appearance, not the content of the image that is important. Since imaginal data can currently be printed at Xerox only on low speed, low volume printers and because of distribution volume, the Newsletter contains a copy made on the 4000 copier (with resultant image degradation). It is appended at the very end rather than in its usual place following this introduction. An original (from the Slot-3100) is being sent to each site for posting.

The second paper, by Paul Roetling, *Visual Performance and Image Coding*, presents data on the eye's ability to resolve what it sees and develops guidelines for determining how much information must be captured and retained in digital form to meet the eye's requirements. Continuous tone images contain a huge amount of data; retaining more than necessary severely impacts storage requirements and processing times.

Paul has written several other papers of a tutorial nature that contain continuous tone images. Check the library for the published versions because the image quality suffers in reproduction. The first describes several methods of representing continuous tone images, the second is a specific method for improving the appearance of halftones, and the third a discussion of factors affecting halftone image quality.

Binary Approximation of Continuous Tone Images, Photographic Science and Engineering, Vol. 21, No. 2, March/April 1977.

Halftone Method with Edge Enhancement and Moire Suppression, Journal of the Optical Society of America, Vol. 66, No. 10, October 1976.

Analysis of Detail and Spurious Signals in Halftone Images, Journal of Applied Photographic Engineering, Vol. 3, No. 1, Winter 1977.

Paul's paper, which originally appeared in SPIE/OSA Vol. 74 (1976) Image Processing, had to be rekeyed and the graphs redrawn for inclusion in the Newsletter. The editor accepts sole responsibility for its appearance and accuracy.

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC]<AltoDocs>WAWnews.press or may be obtained from the editor, Frank Ludolph, XEOS, by messaging <Ludolph> or calling Intelnet 8*923-4356.

M - is used to show the area selected by the mouse (see below).

Selection by mouse

If, while the image is displayed, you point the cursor to one corner of an area you wish to see, depress the left mouse button, move the cursor to the diagonally opposite corner of the area you wish to see displayed, and release the left mouse button, a box will appear surrounding the area you have just selected. If you wish to change the selection, simply repeat the process and a new box will appear after you release the left button. If you try to select an illegal area, for example, outside the area shown, no box will appear and any previous one will be erased. If there is a box on the screen at the time M is keyed, the area outlined by the box is used to define the new window both for the display and for writing into the window parameters in the menu.

In our limited use of this program, we have found this last feature to be very useful. A file name is selected, and the full picture is shown initially. We then use the mouse to select the area we wish to see, key an M to see the region, repeat the process to window down further, until we get exactly the area we want. We then use carriage return to return to the menu and copy down the window parameters for use in other programs as for example, magnify.

The primary purpose of this program is to allow us to experiment with user interaction both for image display and window selection. Thus, we expect that portions of the program will be revised frequently as we conduct experiments. We would also appreciate comments from users on their reactions to program features. Please contact either Keith Knox or myself with comments.

PGR/sm

c: JEBollman
TMHolladay
JEstinehour
LBHolt
DEDamouth
WMReilich
LDMailloux
KTKnox

VISUAL PERFORMANCE AND IMAGE CODING

Paul G. Roetling

ABSTRACT: Sample spacing and quantization levels are usually chosen for digitizing images such that the eye should not see degradations due to either process. Sample spacing is chosen based on the resolution (or high frequency) limit of the eye and quantization is based on perception of low contrast differences at lower frequencies. This process results in about 8 bit/pixel, 20 pixel/mm digitization, but, being based on two different visual limits, the total number of bits is an overestimate of the information perceived by the eye. The visual MTF can be interpreted in terms of perceptible levels as a function of spatial frequency. We show by this interpretation that the total information perceived by the eye is much less than 8 bits times the number of pixels. We consider the classic halftone as an image coding process, yielding 1 bit/pixel. This approach indicates that halftones approximate the proper distribution of levels as a function of spatial frequency; therefore we have a possible explanation of why halftone images retain most of the visual quality of the original.

I. INTRODUCTION

In this paper we consider the problem of how visual performance characteristics can be related to the average number of bits per pixel (picture element) in a sampled and quantized image. If we establish an average number of useful bits per pixel in the sense that only these are used by the eye, we then have a number against which to compare the efficiency of various image coding or bandwidth reduction schemes for cases where system performance is related to the visually perceived image quality.

It is well-known that visual performance is a function of spatial frequency. That is, at high spatial frequencies we do not see as well as at lower spatial frequencies and, without magnification, we can see no detail at spatial frequencies above about 10 line pairs per millimeter. By examining how visual performance varies as a function of spatial frequency, we should be able to establish guidelines for image coding experiments as to which pictorial information is useful to the eye.

In the following sections we will examine the selection of sampling interval and quantization levels based on visual performance, thereby establishing design guidelines and a useful number of bits per pixel as a function of sampling interval. We then examine a simple halftone binary-image code and a similar code applied to multilevel images and estimate the performance of these simple codes on the basis of established limits.

II. SAMPLING AND QUANTIZATION CHOICES

To bound our problem we first assume that the images will be examined at unity magnification at normal reading distance. We lose no generality by this assumption, since all results can be scaled by the magnification. We assume ideal sampling, that is, we ignore sampling aperture effects and display spot (Kell factor¹) effects. We can then assume that if we sample at an interval Δ we can adequately represent pictorial information to a spatial frequency $(2\Delta)^{-1}$.

A simple example serves to show how we use visual performance limits in everyday estimates of coding efficiency. Let us consider the case where the image is sampled 40 times per millimeter. By our assumptions, and the fact that the eye does not see detail beyond 10 line pairs per millimeter, we lose no visible information by taking alternate samples (or perhaps averaging every pair of samples). We have therefore reduced the total number of

samples in the two dimensional image by a factor of four without losing perceived image quality. Most of us would argue that this four-to-one bandwidth reduction should have been obvious, because the information eliminated by the bandwidth reduction was information not being used by the visual system. Thus, we tend to judge the bandwidth reduction in terms of the visually useful information in the original.

To count the useful bits in a picture, we would certainly limit the sampling interval to approximately 20 per millimeter. Similarly, we would limit the quantization levels to about 8 bits (256 levels) since we know that the eye cannot perceive more levels in the output picture. Nevertheless, if we were to multiply 8 bits per pixel by the number of pixels (20 per millimeter sampling interval) we get a considerable overestimate of the number of useful bits in the image. We have failed to take into account how visual performance varies with spatial frequency. In different words, we have established the sampling interval based on the high frequency performance of the eye and the quantization levels based on the low frequency performance of the eye, without taking into account the translation from one to the other.

A VISUAL MTF

We describe an improved approximation to the determination of visually useful bits of information by basing ourselves on known psychophysical data on visual performance in terms of the modulation transfer function (MTF). Text books, such as Cornsweet's², consider various sources of such data. Dooley³ gives a convenient functional form which approximately fits most of the vision data. His fitted curve, in terms of modulation transfer function (MTF), is given by

$$\text{MTF} = 5.05 (e^{-0.138f}) (1 - e^{-0.1f}). \quad (1)$$

This curve has been normalized and f is spatial frequency in cycles per degree. The peak of the curve represents a just detectable modulation of 0.005. The psychophysical data to which this curve was fitted were measurements of the just detectable modulation of a sine wave as a function of the spatial frequency of that pattern. It should be noted that we do not use the data in MTF form, but rather we determine directly the detectable contrast, i.e., we use the original form of the experimental data.

We now assume, for our approximation, that at every spatial frequency we should quantize levels such that the just detectable modulation represents one quantization step. If we further assume that, for all sine waves, the average luminance was mid-grey, then the modulation can be written

$$M = \frac{\text{MAX} - \text{MIN}}{\text{MAX} + \text{MIN}} = \frac{\text{detectable difference}}{\text{total range}} \quad (2)$$

In other words, the number of intervals is the reciprocal of the just detectable modulation, and the number of levels we must represent is the number of intervals plus one. That is, we can write the number of detectable levels as a function of spatial frequency by taking out the normalization from the given MTF curve and adding one, or

$$\# \text{ of Levels} = 1010(e^{-0.138(5\nu)}) (1 - e^{-0.1(5\nu)}) + 1 \quad (3)$$

To convert from the cycles per degree measurement common to psychologists to the cycles per millimeter more common in image processing, we have used a conversion factor of one cycle per millimeter equals five cycles per degree. This curve has a peak at a spatial frequency of approximately one cycle per millimeter. There is evidence³ that threshold measurements are not a good measure of visual performance for all contrasts at frequencies below one cycle per millimeter. To obtain a conservative estimate, we have used Eq. (3) only above one cycle per millimeter, using its maximum value below that point. This curve is shown in Fig. 1.

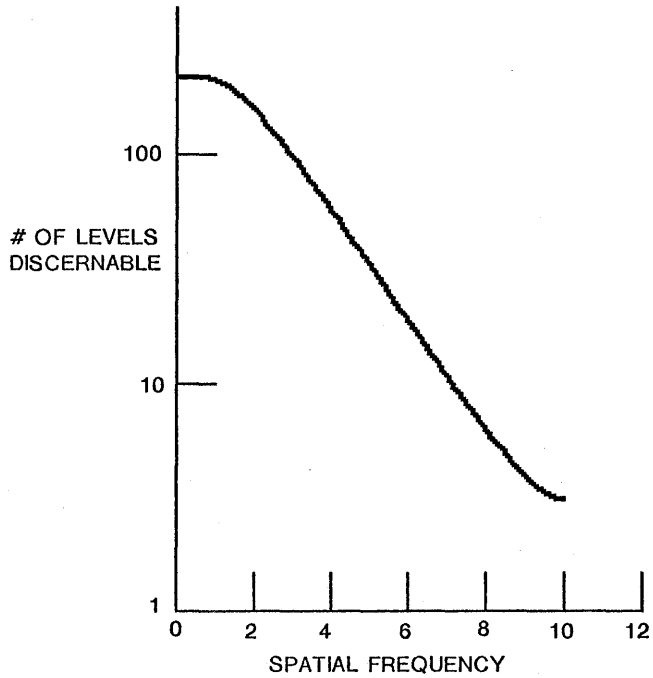


Fig. 1. Visual Performance Limits

B. VISUALLY USEFUL INFORMATION

We can now express visually useful information in a sampled and quantized picture as an average number bits per pixel, by utilizing the visual performance curve in Fig. 1. We take an image area L by L , sampled at an interval Δ by Δ , as shown in Fig. 2a.

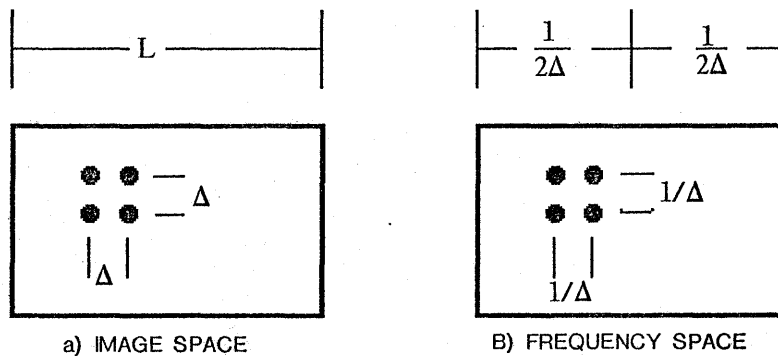


Fig. 2. Image and Frequency Sampling

The visual performance data is described in spatial frequency space. We therefore convert to the image transform space as shown in Fig. 2b. The defined transform space has a frequency range $\pm 1/2\Delta$, with frequency samples spaced at intervals $1/L$. In each space, the total number of samples is the same, that is, n^2 equals $(L/\Delta)^2$. To arrive at an average number of bits per pixel, we evaluate the total number of useful bits and divide by the total number of pixels. In frequency space, we arrive at the total number of useful bits by

multiplying the number of bits per frequency sample ($\log_2[\text{\#LEVELS}]$) by the number of samples per unit frequency interval (L^2), times the frequency interval, and integrating over the frequency range. These operations are combined and expressed as

$$\text{\#bits/pixel} = \frac{1}{n^2} \int \int \{\log_2[\text{\#LEVELS}(\mu, \nu)]\} (L^2) d\mu d\nu \quad (4)$$

where μ, ν are spatial frequencies. Putting together the above relations between constants and the fact that the visual performance data has approximate circular symmetry we can rewrite the expression as

$$\text{\#bits/pixel} = 2\pi\Delta^2 \int_0^{\infty} \nu \text{MAX}\{\log_2[\text{\#LEVELS}(\nu)]\} \nu d\nu \quad (5)$$

Finally, for any high enough sample rate, we can perform a numerical integral approximating Eq. (5) which will give

$$\text{\#bits/pixel} = 2\pi\Delta^2(177.5) \quad (6)$$

If we now insert Δ as approximately 20 samples per millimeter we find that the average visually useful information in an image is approximately 2.8 bits per pixel, substantially below the 8 bits per pixel which we would have estimated had we not taken into account the fall-off of visual performance with spatial frequency.

It is also of interest to note that if we decrease the sampling interval, that is sample more often, the useful information per pixel drops with the square of the sampling interval. We can therefore also calculate a sampling interval at which the average information per pixel would be approximately 1 bit. Substitution in Eq. (6) yields 1 bit/pixel at a sample interval of 33.4 per mm. This result states that, at that sampling interval, an efficient binary code might be found which could represent the image at no loss of visual quality. We therefore consider next simple binary image codes.

III. HALFTONE CODES

One simple form of binary image code has been used for somewhat over a century⁴ by the graphic arts industry. The binary images are referred to as halftones, and are used to give the appearance of grey while printing only full black and white. Such images are generated by combining a non-image related pattern (called the halftone screen) with the pictorial data by addition or multiplication. The combination is then subjected to a threshold to turn the continuous tone to a binary image. This process has been applied to sampled imagery by many authors (for example, see Ref. 5) and the image detail for given screen patterns has been described.⁶

We now ignore the design of the halftone screen to consider what optimal encoding might achieve. The halftone process is essentially one of trading off grey scale for texture. That is, a combination of black/white spots is generated which, when averaged over some area, give the illusion of various shades of grey. Again considering ideal sampling, we examine how a given periodic structure can be represented. Clearly, we must have available at least one sample for every half wavelength of the pattern to be represented. Fig. 3 shows a small region of the picture with two regions identified, each a half wavelength of the desired sample in length. To represent the sample of minimum modulation achievable at this spatial frequency, one bit must change between these two areas of image. This is equivalent to asking how many different levels can be represented by turning on different numbers of bits within the image area whose length is one-half wavelength for the desired spatial frequency. This can easily be seen to be

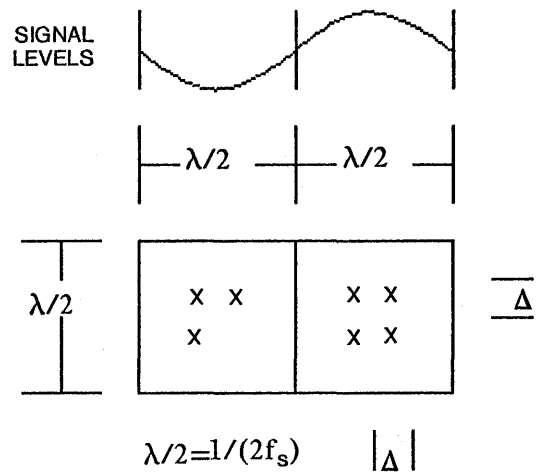


Fig. 3. Image Areas Available to Encode A One Grey Level Change

$$\# \text{ of LEVELS} = (1/(2F_s))^2 (1/\Delta^2) + 1, \quad (7)$$

where f_s is the object frequency, and Δ the sample interval. It is convenient to immediately generalize this result by asking what would happen if each pixel could take on more than black or white values. If each pixel is described by m bits, rather than 1, each element now adds $2^m - 1$ additional non-black values. Thus the same type of coding applied to a multilevel image yields the possibility of representing additional grey levels, given by

$$\# \text{ of LEVELS} = ((2^m - 1)/(2f_s \Delta))^2 + 1. \quad (8)$$

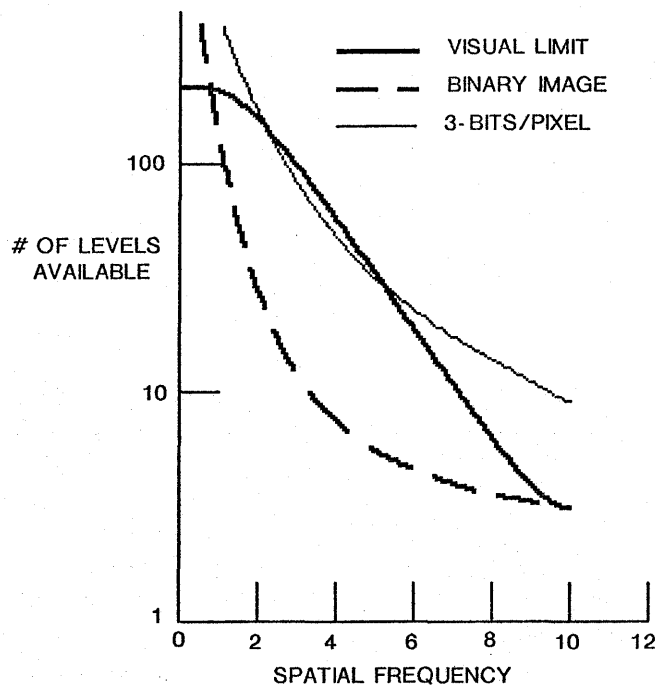


Fig. 4. Texture Code Limits

Fig. 4 compares the original curve from Fig. 1 with the number of levels achievable as a function of spatial frequency for the binary case with a sample rate of 20 per millimeter and for the case of 3 bits per pixel at the same sample rate.

It is interesting to note that the 3 bit per pixel curve indicates that a texture type code should be able to represent almost all information visible to the eye. Since we found that the actual useful information averages approximately 2.8 bits per pixel at this sample interval, we have an indication that the texture type code could be reasonably efficient. In the binary case, at this sample interval, we see that the 1 bit per pixel code falls short of that needed to avoid visible degradation of the image. The manner in which the curve falls with spatial frequency, however, gives an indication of why halftoned images look as good as they do, since the curve shapes are somewhat similar.

IV. CONCLUSION

We have described an approach in which visual data for modulation transfer function of the eye can be utilized to determine the useful information in an image. At a sample interval of 20 samples per millimeter, we have found that the visually useful information corresponds to approximately 2.8 bits per pixel. The shape of the visual performance curve indicates that more levels need to be represented at lower spatial frequencies and less levels at higher spatial frequencies. Thus, it has been shown that halftone or texture codes, although simple, represent image information in a manner which tends to be compatible with the characteristic of the visual system.

V. ACKNOWLEDGEMENTS

The author wishes to express his gratitude to his co-workers for many useful discussions which have helped to clarify the concepts described in this paper. In particular, D. Kermisch and K. Knox provided many helpful suggestions.

REFERENCES

- ¹Costigan, D. M., *Fax*, Chilton Book Co., Philadelphia, Pennsylvania 1971, p. 127.
- ²Cornsweet, T. N., *Visual Perception*, Academic Press, New York, New York 1971, p. 330-342.
- ³Dooley, R.P., "Predicting Brightness Appearance at Edges Using Linear and Non-Linear Visual Describing Functions", presented at SPSE Annual Meeting, May 14, 1975, Denver, Colorado.
- ⁴*Pocket Pal*, Tenth Edition, International Paper Company, New York, New York, 1970, p. 17.
- ⁵Klensch, R. J. "Electrically Generated Halftone Pictures," *RCA Review* p. 517-533 (September 1970).
- ⁶Kermisch, D. and Roetling, P.G., "Fourier Spectrum of Halftone Images," *Journ. Opt. Soc. Am.* 65: p. 716-723 (1975).

Whole ALTO World Newsletter

Technology and Tools

XEROX

APRIL 30, 1978

SPECIAL ANNOUNCEMENTS

WHOLE ALTO WORLD MEETING - The next Whole Alto World meeting is scheduled to be held from 9AM to 3:30PM on Thursday, June 1, 1978, at El Segundo in the Executive Dining Room. Dick Sonderegger and SDD are hosting. The last page of the Newsletter is a flyer announcing the meeting. Please detach your copy and post it on an appropriate bulletin board. *Start your trip to the National Computer Conference right by first attending the Whole Alto World meeting.*

GENERAL NOTES

EIA BOARD LOAN REQUEST - ASD needs the use of an EIA board for software development until July 1. If you know of one that can be spared for a time, contact John McNeley at Intelnet 8*823-2011. It would be greatly appreciated.

THE ALTO USER'S PRIMER - A new document has been prepared for the new Alto user (and the experienced user too) by Frank Ludolph, the Whole Alto World coordinator. It presents, in a non-technical fashion, an overview of the Alto network, hardware and operation, and provides pointers to the documentation. It is included in the Newsletter in the TECHNOLOGY section.

ADDRESS LABEL FORM - Have you wondered about the label used to mail the Newsletter to you? The mailing labels are maintained with Bravo using a form made up by Barbara Baird. The form, complete with instructions, positions each address in just the right place for the Xerox gummed labels, 3R311 (three across by eleven high). Retrieve [MAXC]KForms>Form.AddressLabel, use it according to directions, print on a Dover or Sequoia, and copy onto the label sheets using a standard copier, e.g. 3100, 4500, or 7000. Yes, the Dover should handle the gummed labels, but the printer is a shared resource and your labels might be used to print someone else's output.

WHOLE ALTO WORLD DOCUMENTS - This is just a reminder that the Whole Alto World maintains several documents to simplify your quest for knowledge. The *Alto User's Primer*, in addition to serving as an introduction to the Alto World as mentioned above, serves as a first level index to other, more specific, documents. The *Subsystems Catalog* provides both alphabetical and functional cross reference listings to all the released Alto subsystems, briefly describing each program and pointing to its documentation. The *Hardware Catalog* serves a similar function for the various hardware components. The *Alto Network* drawing maps on a single sheet of paper the Ethernets, Gateways, and servers that comprise the network. Lastly, the *Newsletter* provides a monthly look at changes in the Alto environment. Each of these documents, revised periodically, is maintained on the AltoDocs directory of your local IFS or MAXC.

TOOLS

HARDWARE

EIA BOARDS - A printed wiring version of the EOS-developed EIA board is being built by ASD. The price will be based on the number of units built. Contact Frank Brinkerhoff at Intelnet 8*823-1096 by June 1. There has been a great deal of interest in this board since the original EOS build closed; don't miss this opportunity.

MAGNETIC TAPE CONTROLLER - A mag tape controller for the Alto II is now being designed by ASD. It will drive a 1600 BPI transport such as those built by Kennedy. A build is being scheduled for delivery in the August-September timeframe. The estimated cost for *controller and drive* is about \$7K (actual price will be based on quantity and will include the sharing of engineering costs). Liz Bond is coordinating the orders for Hoag Nielson. Contact her at Intelnet 8*844-1064 to order a controller. Kennedy drives, which must be ordered separately, may be obtained through Versatec at EOM prices.

THE TRIDENT QUIET BOX HAS ARRIVED - The quietbox has been installed and evaluated. Ted Stollo reports:

"Jensen Engineering of Santa Rosa, Calif has made up a prototype of a T80 quiet box. The PARC purchasing department is likely going to go out on competitive bids on these since there are about 500 or more T80 units in Xerox. PARC will attempt to negotiate a master agreement for such units. You should direct any inquiries to Clay Osterhout at PARC. For those of you with immediate needs, you can contact Jensen direct via your own purchasing departments. Jensen is quoting prices of \$655 for single units, \$589.50 for 10-24 units, \$556.75 for 25-49 units, \$543.65 for 50-100 units. Contact Harold Jensen at 707-544-9450. Delivery has been quoted as 30 days.

"The unit comes in a color scheme which matches that of Alto II. It is sized to match a companion enclosure for the T300. This makes the unit relatively large for just a T80. A storage compartment is provided inside the base with precautions having been taken to minimize storage interfering with air flow for cooling.

"Our experience with the prototype unit has been quite good. It is very quiet. In fact, the ambient room noise(61DBA) from air conditioning and Alto fans is so high that we cannot measure the S/N improvement precisely but it is definitely more than 23db. We have had the unit for two weeks now and have specified the following improvements for the final product all of which have driven the price up from the original estimate.

- 1) Since the T80 unit is so heavy we have gone to a revised way of inserting the unit into the box which involves removing a top cover (by removing 4 screws and a hinge bar). Also the T80 unit can be removed leaving its cabling in place in the box. The unit can be serviced in the box by lifting this cover.
- 2) We required an hydraulic cover closer like that used in the Dover to prevent covers from crashing down on top of unit.
- 3) A vibration mode was discovered in running the T80 diagnostics which caused the unit to move back and forth on its casters. We therefore have required heavy duty leveling feet in addition to the casters.

- 4) Temperatures inside the quiet box close to the T80 side skin have been observed to get as high as 95.3 degrees F. The exhaust fan air temperature is 97 degrees F vs a free standing T80 exhaust temperature peak of 99 degrees F. While the unit has run error free on diagnostics and in use with the Press software, we feel the temperature must be kept a little lower. Calcomp spec calls for a maximum ambient of 100 degrees F. We are requiring the addition of another air intake fan at 70 CFM, low noise to further cool the unit. The prototype has one such intake fan and one such exhaust fan.
- 5) We have required hinge modifications from the prototype which will permit units to be butted side by side next to each other.

"I am not specifically endorsing this unit. It is important to give some weight to the purchasing dept's needs of getting competitive bids, negotiating a master contract with someone. We do not want to be caught in the position we were with our mouse supplier where price escalation was out of sight due to our lack of foresight in procuring the units originally."

MAINTENANCE NOTES

CATCHING PARITY ERRORS - Occasionally users will experience memory parity problems although DMT reports no errors. While DMT will catch bad memory chips, it will have trouble detecting chips that fail intermittently or only with certain bit configurations. One method of catching these intermittent chips is to record the location and content of parity failures when displayed by SWAT. Note the common bits set by anding together the contents of all even or odd address words within a given 4K address space, e.g. 0-17777. (Use a 16K space for the XM machines). Using the memory layout maps attached to the Newsletter, map the common "on" bit to a specific chip and replace it. Of course, this may not always reduce to a single chip, but the number of possibilities is greatly reduced.

If the system continually crashes running different subsystems without first reporting a parity error, it could happen that the error is in an area occupied by a vital part of the operating system. Try toggling the Memory Configuration Switch to the alternate position which reverses high and low memory by complementing the high-order memory address bit.

ALTO XM PROBLEM - The 16K memory chips used in the extended memory Alto (and all 7th build Altos) unlatch data sooner than the old 4K RAMS. The Orbit microcode accesses the value twice, always getting zero the second time. The result is that Dovers and Sequoias, which use the Orbit, cannot currently be run with an XM machine. An EO is being generated which causes the data to be latched for 10 microseconds or until the next MAR←, whichever comes first. Alto maintainers have been messaged the wiring changes to take care of things until the formal EO arrives.

ALTO I ETHERNET HARDWARE BUG - A *very infrequently* occurring bug has been located in the Alto I Ethernet interface which causes the system to SWAT with a parity error at location 600. This occurs when a Pup packet is received that is just slightly longer than the receiving buffer (about 2 words or so). What happens is that just as the packet is ending, the microcode will run out of room in the buffer, take an early exit from the input main loop, and dive into the status posting code where it says:

```
mar ← EPloc      ; = 600
```

...
md ← EPFct ;gate interface status to bus.

Unfortunately two of the interface status bits, CRC and IT, are *not synchronized*. They can be changing even as they are slithering down the bus into memory, causing the parity bit to have an indeterminate value. (Alto IIs have a register between the processor bus and the memory which serves as a synchronizer, so they aren't susceptible to this bug.)

The next microcode will contain a fix, changing the status posting microcode to run the status through an R register before putting it in memory, thus using the register as a synchronizer. Since 600 is such a magic number, it would have been noticed before in the last 4 years. Since it hasn't, it's not worth changing the ROMs in every Alto I, but the fix will be there in the microcode if a change is made for any other reason.

HARDWARE CHECKOUT BY THE USER: PART II - Last month, methods of verifying performance of the Alto's workstation using CRTTEST and KEYTEST were presented. This month the diagnostics used to checkout the Alto processor are discussed. DMT and MADTEST are used to test the various functional pieces of the Alto processor: main memory, microprocessor memories (RAM and PROM), arithmetic-logic unit (ALU), registers, and data paths (it isn't necessary to know what they do).

DMT tests the main memory, the area occupied by your data and most, if not all, of the subsystems that you run. When DMT is executed, the display will be black except for a small white square that bounces randomly about the screen.

DMT should be run for long periods, say overnight or over the weekend. If the Alto is left in the Executive, DMT will be called after 20 minutes. In the morning, while waiting for the disk to spin up to speed, depress and hold the 's' key. A three line message will be displayed a few inches from the bottom of the screen. The second and third lines should begin "0 Errors...". If some errors have been found, the memory chip location(s) will be indicated. *Don't boot the Alto; the machine may not work and the location information will be destroyed.* Inform the local maintenance group.

MADTEST, the Microcode Alto Diagnostic Test, exercises much of the rest of the processor. It is actually a collection of routines that test the RAM, PROM, ALU, registers, and data paths.

It is run in the same manner as KEYTEST and CRTTEST, namely, execute NetExec by booting from the Ethernet or typing netexec^{CR} to the Executive, and then type madtest^{CR}. The screen will indicate at the top the test routine being run, the number of passes completed, and the errors detected (if any). The middle of the screen will contain trash (the unformatted contents of memory) and below that, a band containing the phrase "Black on white means DO TEST". Immediately below the band are the various tests that will be run.

The cursor consists of the usual arrow (like Bravo) with a changing series of black and white horizontal lines through it. The number and location of the lines change to let you know that something is happening, because it isn't always apparent from the rest of the screen. The cursor also moves about the screen in its usual fashion except that sometimes it jumps (after a few seconds delay) rather than moving smoothly as is customary. You may also notice the screen "tearing". Both of these effects are a result of the level at which the

diagnostics operate.

MADTEST will automatically run all the tests, over and over. The number of passes completed is displayed near the top of the screen; allow MADTEST to run several passes. If any errors are reported, write them down and pass them along to the maintenance people. *Note: depressing the SPACE bar will halt the program and clear the screen. To suspend execution without clearing the screen, move the cursor into the area of the test list. Moving it out will resume execution.* After the test has run several passes, either boot to get to the Executive or type SHIFT-SWAT for DMT.

To prevent any of the tests from being run, move the cursor into the area of the test list and bug the test not to be run. It should change to white letters on a black background. To cause it to be executed, bug it again. Any tests shown in black letters on a white background at the start of a pass will be executed. The default is to execute all tests.

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local IVY server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [XEOS].

NEW RELEASE: ShowAIS - Joe Maleson has provided us, via boot server, a program that halftones and displays 8 bit/pixel AIS files from a remote file server. It is similar in function to AISshow which expects the file to be on the local disk. The documentation from [IVY]Maleson>ShowAIS.bravo is attached to the Newsletter. Note: This program can degrade the performance of the file server. If this seems to be happening, please delay use until the need, by others, of this scarce resource is reduced.

ReReleases - Subsystems

CHAT - Only minor changes were made to this release. It is available from boot servers.

DRAW - The clandestine "Color-DRAW" is now officially released as DRAW 4.0. Load DRAW.dm. A documentation update is available on DRAW-new.press.

PRESSEEDIT - Only minor changes were made. Retrieve PressEdit.run.

PUT - Version 1.1 has been enhanced to work on disks using the 'big disk' feature of the new operating system. People with model 44 disks who use 'big disk' will need the new PUT. Retrieve PUT.run.

READPRESS - The floating point spline description is now printed out in decimal floating point rather than its octal representation. Retrieve ReadPress.run.

TRIEX - The latest version runs on Alto's successor machines and also has a few bug fixes. Retrieve Triex.run.

ReReleases - Packages

MICROFLOAT.DM - Double precision floating load of the value zero now functions properly. Load MicroFloat.dm.

TECHNOLOGY

This month, instead of the usual technology paper, we present a new, introductory level paper for the new and not-so-new Alto user. Most existing documentation tells a person how to use a specific item. By contrast, this paper describes, in non-technical terms, the system's existing facilities, provides an overview of the documentation scheme, updates some basic operational procedures and documents some procedures that are unrecorded elsewhere.

Of specific interest to current users are the documentation scheme overview in chapter 3 and the machine check-out procedures in chapter 4 (which were serialized in these last two editions of the Newsletter).

This document, along with a copy of local procedures, should enable the new user to get started in a shorter time and with a minimum of frustration.

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC]<AltoDocs>WAWnews.press or may be obtained from the editor, Frank Ludolph, XEOS, by messaging <Ludolph> or calling Internet 8*923-4356.

THE ALTO USER'S PRIMER

BY FRANK LUDOLPH

APRIL 1978

XEROX WHOLE ALTO WORLD

INTRODUCTION

I. THE NETWORK

- Alto
- Ethernet
 - Transmitting data
 - Receiving data
- Gateway
- Servers
 - Printers
 - File Storage
 - Electronic Mail
 - Boot Files
 - Time
 - Device Name List
- Network diagram

II. THE FILING SYSTEM

- Alto Filing System/Filenames
- File Servers

III. WHAT YOU NEED

- Documents
 - Overview
 - Basic Document List
- Subsystems
- Accounts
- Keeping Up-To-Date

IV. OPERATIONAL PROCEDURES

- Operating the Alto
 - On/Off
 - Loading/Unloading a disk
 - Booting
- Checking Out Your Machine
 - Workstation
 - Disk Drive
 - Alto Processor
- Using the Executive
 - Starting a service
 - Correcting typing errors
 - Aborting a service
- Initializing/Installing a New Disk
 - Copying
 - Using the network
 - Installing a new disk
 - Retrieving a basic set of files
- Normal Disk Maintenance
 - Listing filenames
 - Listing file attributes
 - Display a file's content
 - Deleting files
 - Grouping files
- Disk Crash!
- The User Command file
- Printing
 - Press files
 - Bravo files
 - Ears files
 - Gypsy files
 - Text files
 - TTY files
 - Other printers

INTRODUCTION

Most of the existing documentation for the Alto and its systems tell you *how to use* the Alto, Bravo, FTP, etc. The purpose of this paper is to present *what exists* and *where to get it*. It is intended as a first reading for the new user and is primarily non-technical in its presentation. The first chapter, THE NETWORK, presents a brief overview of the system's services, components, and their interconnection. The second describes the filing systems, both Alto and IVY, and their interaction. The chapter, WHAT YOU NEED, presents an overview of the documentation available, lists the minimal set of documents and subsystems needed to get started and where to get them, and tells you how to keep up-to-date. Lastly, there is a *how to* chapter comprised of operational procedures that either have been revised or are unrecorded elsewhere.

Many of the sections reference documents for further study. While many of them should be available from your local file server, the original source is indicated. *It is advised that these papers not be read until reading of the Primer is completed and fully understood.*

Many sites have established local procedures to cover establishing accounts and methods of obtaining and initializing disks. Ask other machine users in your group or area to direct you to the appropriate people. In case you find yourself all alone or no one seems able to help you, call the Whole Alto World coordinator. Currently the coordinator is Frank Ludolph. You can reach him at Intelnet number 8*923-4356 or sending by a message to <Ludolph>.

A word on protecting Xerox information. The systems you are using are prototypes for future Xerox products. Much of the technology involved cannot be covered by patent; your discretion is required in protecting these developing technologies from premature disclosure. While the corporation has publicly indicated its intention to market a broad range of Office Information Systems products, it has been mute on exactly what form they might take. Telling your friends about the systems you use may damage Xerox's profitability based on these technologies by either prematurely indicating trends in future products or preventing the patenting of a process. Don't chance diminishing the success of future products by your indiscretion.

After reading the Primer, it is suggested that you acquire the basic set of documentation and accounts, find an Alto, and get some hands-on experience. The Machine Operation section in this paper and the Alto User's Handbook can be used to build yourself a disk. Then tryout FTP, the file transfer program, and Bravo, the text editor. Next use a couple of the graphics programs, Draw, Markup, and Sil. Create some documents and print them; these are all functions that you will probably use no matter what else you do. Now that you're fairly conversant with the Alto, look through the Subsystems Catalog for other subsystems that you might want to use, retrieve the documentation and programs and try them out. You're off and running. Enjoy.

I. THE NETWORK

The network is composed of Altos and other computers connected to several geographically dispersed, technologically innovative local computing nets which, in turn, are tied together by minicomputers over standard leased and/or dial-up telephone lines. The local computing nets are called Ethernets; the mini-computers linking them are referred to as Gateways, the latter providing several services in addition to linking together Ethernets. The last page of this chapter is a diagram of the network as it is currently implemented.

THE ALTO

The Alto is used to prepare and print documents containing both text, diagrams, and images, convey messages electronically, aid circuit and IC design, and, of course, write programs. It is a minicomputer consisting of a processor, disk drive, workstation, and Ethernet transceiver. The microcoded processor has 64K of 850ns, 16-bit word semiconductor memory (extended memory versions are available). A 1K microinstruction RAM can be loaded with special purpose microcode to extend the instruction set, perform special functions or drive special I/O devices. It is packaged with the disk drive and power supplies in an under-table-size cabinet for easy placement in the user's office.

The disk drive commonly supplied with the processor is the Diablo Model 31 disk drive, though other Diablo models can be used. The Model 31 accepts a single disk which can be used to store about 2.5 megabytes. The average seek time is 70 ms, the average transfer rate, 1.22 MHz.

The workstation is composed of a vertically oriented video display which is refreshed 30 times per second, a standard keyboard with a few extra keys, a mouse (pointing device), and five-finger keyset. The display is composed of 808 lines, each line having 606 individual points (a 606 by 808 bitmap). Each point can be individually controlled to produce not only text but also graphics and even pictures. The cursor, whose position is controlled by the mouse, is a 16x16 bitmap whose shape is under program control, independent of display content.

The Ethernet transceiver connects the Alto to the Ethernet, described below. Using the Ethernet, the Alto can communicate with a large number of other Altos, computers, and special purpose servers.

See: *Alto User's Handbook* and [MAXC]<AltoDocs>Subsystems.press, Executive section (operation); [MAXC]<AltoDocs>AltoHardware.press (hardware description).

THE ETHERNET

The Ethernet was designed to provide a highly reliable communication facility for computers located within a single building or small complex. It consists of a single length of coaxial cable (the Ethernet) and transceivers that connect each computer to the cable. Each computer on the Ethernet has a unique three digit octal address and the Ethernet itself has a unique two digit octal address. These addresses are wired on the Ethernet board in the Alto. Data sent over the Ethernet is sent in packets, each packet containing the net and machine addresses that identify the source and destination machines.

TRANSMITTING DATA Rather than controlling access to the cable by a single active element whose failure would result in the loss of the communication channel, control is distributed among the transceivers and communicating programs. Each transceiver, specially designed to prevent contamination of the Ethernet in the event of failure, verifies that the cable is clear before transmitting and listens during transmission for interference, retransmitting at a randomly determined later time if interference is detected. Each packet of data transmitted is assumed to have only a *high probability* (as opposed to certainty) of reception; it is the communicating programs' responsibility to verify receipt of data and retransmit if necessary.

RECEIVING DATA The interface module checks each packet as it passes for its own net and machine addresses. Packets with matching addresses are buffered and passed to the executing program. The program will normally acknowledge receipt of the packet by transmitting to the sender an appropriate reply. The receiving program should be prepared to discard duplicate packets (the sender may not have received the acknowledgement and retransmitted the packet, assuming it was lost).

See: Ethernet: *Distributed Packet Switching for Local Computer Networks* by Robert Metcalf and David Boggs. Parc Universal Packets (PUP) are described in several papers on [MAXC]<PUP>, especially PUP.ears.

THE GATEWAY

The Gateway's primary function is to transmit packets between Ethernets. It is a small computer (currently Novas but soon to include Altos) that attaches to an Ethernet in a manner similar to any other computer except that it responds to all packets addressed to other nets. Each off-net packet is picked up, encapsulated, and routed over the appropriate lines to other Gateways. Transmission of data packets through Gateways is transparent to the sending and receiving programs; packets, after passing through Gateways to the destination Ethernet, are identical to the original packet;

More than one Gateway may be attached to an Ethernet and there may exist more than one path between two Ethernets, that is, there may be loops forming a true network structure. (Remember that it is the responsibility of the communicating programs to recognize and discard duplicate packets.)

The Gateway also provides bootfile, time, and name look-up services. These functions are described below.

See: [MAXC]<PUP> for some papers on Gateway protocols.

SERVERS

Servers provide services that can be provided more effectively in a centralized fashion. In general these services require special hardware, use intermediate facilities to provide around-the-clock access, extend local facilities (file storage), or provide backup.

PRINTERS A printing server consists of an Alto, a printer, and usually some additional disk storage (a second Diablo 31 or a Trident T-80). Like all other Altos on the network it has an address and usually a name (see the Ethernet Description above). To send a file to a printer, the printer's name or address must be indicated. To save you the extra keystrokes of entering the printer's name or address everytime, a default printer can be specified in the [HARDCOPY] section of the User.cm file. (See the PRINTING and USER COMMAND FILES sections under OPERATIONAL PROCEDURES). Once the User.cm file has been modified, you need to specify a printer's name or address only when you wish to use a different printer.

The most common printers are Spruce printers (Spruce is the name of the software that runs on the printer's Alto). They will print Press format files containing text and graphics, e.g. line drawings, of limited complexity. (For directions on how to send a file to a Spruce printer see PRINTING). The server listens to the Ethernet, waiting for a print request. When a request is detected, the file to be printed is transferred as soon as possible to the printer's disk (but not until the printer has finished printing the current file). These spooled files are then converted, in the order received, to a format suitable for printing and printed.

Spruce is normally run on two types of printers: Dover and Sequoia. Dover is a high volume printer, its output looking very much like a good clean xerographic copy. Sequoia has better solid area development, that is, the blacks are blacker, but is intended for low volume (not over 200 sheets at a time). Dovers are generally available 24 hours a day, Sequoias only during normal working hours (due to limited component life). Your User.cm should probably specify a Dover, if possible, as your default printer.

Press printers, i.e. printers running Press software, will print not only text and graphics of unlimited complexity, they will also print images such as halftoned photographs. Press is rather more complex to use and generally does not run in server mode; you must personally supervise the transfer of the print file and initiate printing (although a form of server mode can be setup when appropriate). Press can be run on any of the printer hardware but is normally run only for experimental purposes on printers other than Dover and Sequoia. (Images cannot be printed on Dover by Press because of the Dover's high speed).

See: [IVY]<Spruce>SpruceManual.press, Subsystems Manual, Empress section, and [MAXC]<GR-Docs>PressOps.ears.

FILE STORAGE The Diablo 31 disk drive that comes with the Alto can store about 2.5 million bytes on a disk. That sounds like a lot but after putting on an operating system and various subsystems, it starts to dwindle rapidly. One solution is to have a lot of disks, but changing disks takes a couple of minutes and sometimes you need a file off another disk. Transferring files between disks can be inconvenient, particularly if you don't have a dual-disk Alto. It can also be kind of scary if you have only one electronic copy of a file; in the admittedly unlikely event of a disk failure, you may have to completely re-enter or recreate the file, if you can.

The solution to these and other problems is a file server, IVY (or IFS), usually one per geographical site. An IVY station consists of an Alto and one to eight Trident T-80 and/or T-300 disk drives, each unit having 80 and 300 megabytes of storage respectively.

An IVY account, obtained from the local IVY administrator, will provide you with an additional storage as needed. With your own account, you will have access to public subsystems and documentation maintained on special directories, and you can freely move files between the Alto disk and your IVY directory. Keep files on IVY that you have finished with but wish to keep around for future use. Use IVY as it seems convenient. Don't worry about deleting a file from your disk after transferring it to IVY. Not only is there an excellent program to rebuild Trident disks in the event of a failure, but, at most sites, files are also backed up each night on a second disk for use in the event of a catastrophic failure.

Files on IVY aren't quite as private as on your Alto disk; though others can't write over them unless explicitly given permission, they can read them unless you invoke IVY's protection facilities to prevent it. In any case, others can still list the filenames, protected or not.

There is a second file server on MAXC, a large computer on the Ethernet located at PARC. It is used in a manner similar to IVY but is intended for use by PARC people and others that work closely with them, such as SDD and ASD. MAXC also provides storage facilities for the mail servers discussed below.

See: [MAXC]<IFS>HowToUse.press (user manual) and Operation.press (administrator manual).

ELECTRONIC MAIL has replaced the short hand-written memo for Alto users. Messages handled in this manner are delivered immediately to as many people as appropriate, provided of course that they have a MAXC message account. Message accounts are generally available to members of PARC, SDD, ASD, and others that work closely with PARC. Attempts are being made to establish at least one group account for each Alto-using group in order to give all users access to the message system.

There are two message systems now in use: MSG, which is run from your Alto on MAXC, and Laurel, which runs entirely on the Alto, using MAXC only to hold undelivered mail. MSG is a teletype oriented system with comparatively primitive editing facilities, though it

does provide a wide set of message control operations. All messages, both intransit and received, and the MSG software reside on MAXC.

Laurel is now replacing MSG. The major differences are that it is oriented toward Alto-type operations, such as menu picking and Bravo-type editing, and resides entirely on the Alto disk. Undelivered messages are currently kept on the MAXC file server but they will soon be held on IVY. To receive messages from Laurel you must have a MAXC MSG account, though it does permit people without message accounts to send messages to registered users. MSG and Laurel are essentially compatible; users of each can send messages to the other.

See: [MAXC]<DMS>Laurel.press.

BOOT FILES One of the secondary services provided by Gateways is the boot server. Several standard subsystems have been converted to boot file format. This permits them to be stored on the Gateway's disk and transmitted on request to your Alto. This saves you disk space and permits you to execute those programs even when you don't have a good disk available. Some of these boot files enable you to build a working disk, some are commonly used programs, and some are diagnostics used to verify the operation of your machine.

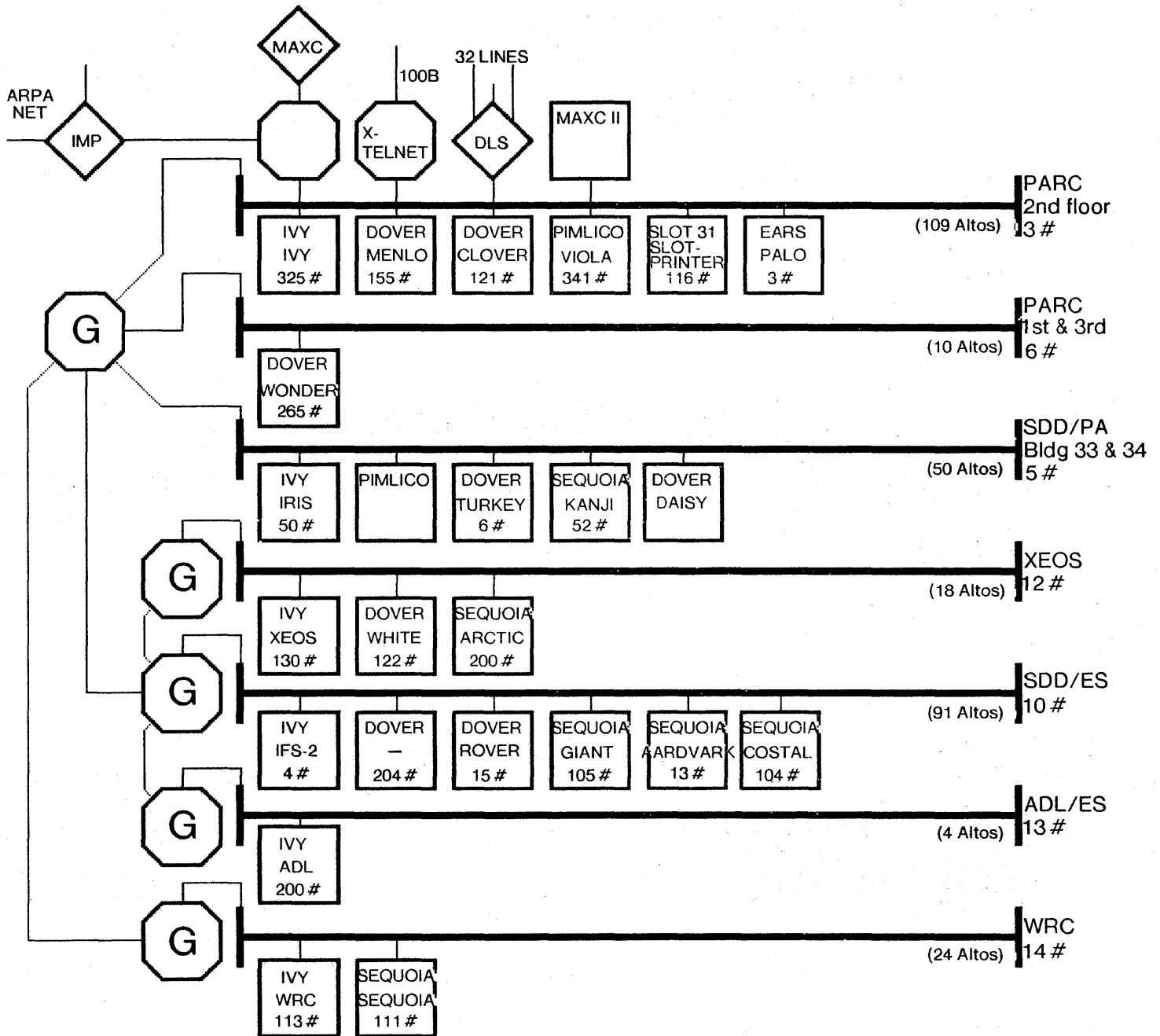
See: <AltoDocs>Subsystems.press, NetExec section.

TIME Your Alto may try to set the time-of-day whenever it is booted. If possible it will get the time from another machine on the Ethernet. Since the Gateway runs 24 hours a day, a time server has been included in the software. Gateways will, in turn, attempt to get the time from other Gateways, so everyone should have the same time, adjusted for their own time.

DEVICE NAME LIST Each device attached to the Ethernet has an address and, probably, a name. Names are provided because they are easier to remember and can be used to provide an indication of the machine's use. Since only addresses are recognized by the Ethernet interface, the address of a destination machine must be substituted for its name before sending a packet. The Gateways maintain a name/address correspondence file and provide a name lookup service so that programs on originating machines can request the address of the user-supplied machine name.

XEROX ALTO NETWORK

APRIL, 1978



Filed on [MAXC]<AltoDocs>AltoNetwork.press

TYPE
NAME
ADDRESS
ALTO Server
(ALTO + Device)

NOVA
(G is w/ pup Gateway software)

Other Resource

II. THE FILING SYSTEM

Files contain documentation, subsystems, or data and are stored on a disk, either the disk in the Alto or the disk of a file server. The subject of files is very important in the operation of the Alto so take the time to understand it completely.

THE ALTO FILING SYSTEM

The Alto stores on its disk all the material with which you work, as well as the programs you use. Each document and each program is stored as a different file. Because the storage space on a disk is finite, programs and the materials they require may be grouped on different disks, for example a document disk, a BCPL programming disk, a MESA programming disk, and a Design Automation disk. A disk will normally function well if there are several hundred free pages. Let experience and the amount of free space on the disk be your guide. Some disk maintenance techniques are contained in the OPERATIONAL PROCEDURES chapter.

A file is identified by its name, a string of letters, digits, and the characters "+-!\$" and is no more than 39 characters in length. Upper and lower case letters can be used interchangeably; they're identical to the filing system. No spaces are permitted.

The filename can have two parts, the *main name* followed by the *extension*, separated by a period. The main name is any group of characters that make sense to the creator of the file. The extension is used in a systematic manner to give people hints as to what the file is for. For example, a program has the extension ".run". Files created by a program often have the program's name at its extension, e.g. Report.bravo was created by the program Bravo.run. In a few cases, the extension indicates the format or intended use of the file; Document.press is a file that can be printed. Common extensions are:

.AL	Alto display fonts
.BR	Precompiled modules ready for loading with others to form a program
.BOOT	Subsystems files that can be executed directly off a boot server
.CM	Command files which the Executive will treat as keyboard input
.DM	Groups of files packaged together for easy transfer and storage
.EARS	Document files ready to print on the Ears printer (Palo Alto only)
.EP	Ears fonts (Palo Alto only)
.PRESS	Document files ready for printing on printing servers
.RUN	Subsystems (programs)
.SYMS	Symbol tables used to debug a program
.TTY	Document files that can be printed on servers

And don't forget all the extensions that reflect the creating subsystem e.g. .BRAVO, .BCPL, .MESA, and .DRAW.

See: *Alto User's Handbook*.

THE FILE SERVERS

Files are also located on the IVY and MAXC file servers (see THE NETWORK). The storage space is broken into chunks, each with its own directory. Some of the directories contain files intended for general distribution, such as subsystems and their supporting documents, while others provide individual users with a place to store files when not required on the Alto disk. To get a chunk for yourself you need to get an account from your local IVY administrator. A file must be on the Alto disk to be used; the file servers are only for storage.

Each of the file server stations has a name, just like all the other machines attached to the network. Many of them are named for the facility or group they serve, e.g. WRC, XEOS, and ADL. The network diagram at the end of the first chapter indicates all the stations.

Names of files on servers are the same as on the Alto disk except that a server needs an additional part, the directory. If you want to tell someone else where to retrieve a file, they must also know the server's name. For example, the Executive subsystem is stored on [MAXC]AltoExecutive.run. Notice that the name of the server is enclosed in "[...]" and the directory in "<...>". When it comes time to retrieve the file, you will connect to the server and identify the file by the directory, main name, and extension. Note that the directory can also be of the form "<...>...>". The name between ">...>" is a subdirectory name and is used to group files in a fashion similar to the extension.

Using a file server you can:

- list the names of files under a directory,
- retrieve subsystem and document files from public directories,
- retrieve files from other directories if the files are unprotected (the default),
- store and retrieve files from your own directory,
- store files under other directories if you know the password for that directory.

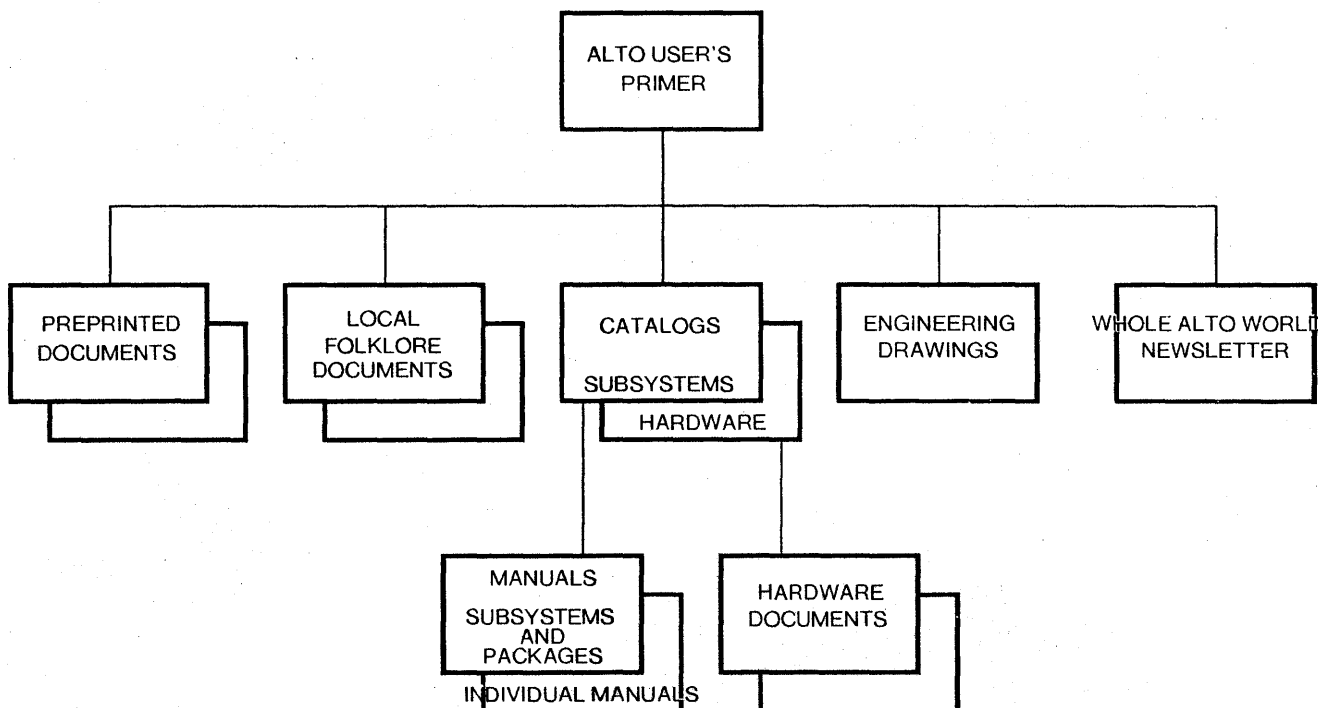
See: <AltoDocs>Subsystems.press, FTP section, and [MAXC]IFSHowToUse.press

III. WHAT YOU NEED

Much of what you will need, programs and their documentation, is stored on file servers. Typically programs and program modules are stored under the <Alto> directory. Documents are placed in the <AltoDocs> directory. There are some exceptions which will be pointed out below. A program, when retrieved onto the Alto disk is immediately usable, though some first must be installed (normally performed by typing program name/i). Documents may be viewed on the screen using Bravo if they have the extensions .bravo or .memo, using Markup if the extension is .press, or with Type, an Executive routine, if in a .tty file. If you want printed documents, see the section on printing under OPERATIONAL PROCEDURES.

DOCUMENTS

One of the most difficult problems when learning to use a new system is finding the existing documentation. A hierarchy of documentation has been created with this document as the root. The overall documentation scheme is illustrated in the figure below.



Several commonly used documents have been printed and bound. These include the *Alto User's Handbook*, the *Bravo Course Outline*, and the *MESA Language Manual*. The Handbook contains introductory material on how to get started on the Alto and manuals for several commonly used subsystems: Bravo, DDS, Draw, FTP, and Markup. Though it was written in late 1976 and is somewhat out of date, it is well done. It is certainly accurate enough for the beginner; most of the changes have been in the form of enhancements or additions.

The *Bravo Course Outline* is a comprehensive teaching and reference guide to using Bravo, a widely used text editor. There is a companion workbook that contains practice material. Both have been recently updated.

Local folklore covers a number of papers written to document local procedures and supplement program manuals for several subsystems. Though intended primarily for use at the site for which they were written, you will find them useful at other sites as well. Folklore documents known to exist are:

ORGANIZATION	FILE(S)
SDD	[IRIS]<SDSUPPORT>SDDocs>* (several files)
	[IFS-2]<SDSUPPORT>SDDocs>* (same as on [IRIS])
PARC	[MAXC]<AltoDocs>ParcAltos.tty
XEOS	[XEOS]<AltoDocs>Folklore.bravo

There are now *catalogs* that list and briefly describe subsystems and hardware. Each entry also references the supporting documentation. In addition, the software catalog crossreferences the subsystems by function to simplify locating a program to perform a given task. They are stored under the <AltoDocs> directory on most servers as SubsystemsCatalog.press and HardwareCatalog.press.

Most subsystems are documented in the *Subsystems Manual*, a compendium of individual manuals. Most of the smaller subsystems are included, only the larger documents are absent. Each of the included *subsystem manuals* is also available individually as a .tty format file. As new versions of subsystems are released, retrieve and print the revised documentation and replace that part of the Subsystems Manual. A similar collection, the *Packages Manual*, contains documentation relating to general purpose program modules.

Engineering drawings, EOs, for the Alto are forwarded to maintenance people by the coordinator as changes are made to the existing hardware.

The *Whole Alto World Newsletter* is a monthly publication for Alto users. It contains information of a timely nature, notes on hardware and new and revised software, and papers on Alto-related technologies. Contact the coordinator if you wish to receive the Newsletter.

BASIC DOCUMENT LIST Most of these documents can be obtained from the local file server (or MAXC) and printed on a Dover or Sequoia. Some of them are not available in electronic form or are not in a format suitable for printing on the local printers. Those documents may be obtainable locally. If not, contact the coordinator, listed in the INTRODUCTION, for help. The following list is not comprehensive, but should be sufficient to begin with.

DOCUMENT	SOURCE
Alto User's Handbook	Local sources
Bravo Course Outline	Local Sources
Subsystems Catalog	<AltoDocs>SubsystemsCatalog.press
Alto Subsystems	<AltoDocs>Subsystems.press
How To Use IFS	<IFS>HowToUse.press
Laurel Reference Manual	[MAXC]<DMS>Laurel.press

If you intend to do some BCPL programming you will need:

BCPL Reference Manual	<AltoDocs>BCPL.ears/.tty
Operating System Manual	<AltoDocs>OS.press/.ears
Alto Hardware Manual	<AltoDocs>AltoHardware.press/.ears
Packages Manual	<AltoDocs>Packages.ears
Gypsy Operator's Handbook	<AltoDocs>Gypsy.press

If you wish to use MESA and are outside SDD, PD, ASD, or PARC, contact the coordinator.

Design Automation System users will need the SIL Manual on [IRIS]<SIL>SilManual.press. It includes documentation on the other programs in the system.

SUBSYSTEMS

When a new disk is built, what subsystems should be put on it? That, of course, depends on what it will be used for, but that normally breaks down into just a few categories: document creation, BCPL or MESA programming, or Design Automation. (Of course there are several specialized functions but if you know about that, this Primer probably isn't necessary.) Almost everyone needs a non-programmers disk (document creation), even if they intend to program, so we'll start there. The disk should contain:

BASIC SYSTEM SUPPORT

EXECUTIVE
 INSTALLSWAT run to put SWAT and SWATEE on the disk
 USER.CM contains user settable defaults for many subsystems

FILE MANAGEMENT

FTP moves files between Altos and a file server
 SCAVENGER restores a flaky disk (if space is tight, use the boot server)
 DDS simplifies disk maintenance
 or PUT for disk maintenance, smaller than DDS but fewer functions

DOCUMENT CREATION

BRAVO text editor
 SIL diagrams of vertical and horizontal lines with captions
 and/or MARKUP diagrams that include diagonal lines and mouse tracks
 and/or DRAW diagrams that include curves
 PRESSEEDIT merges text and diagrams
 EMPRESS sends files to a printer (.press and .tty formats)
 NPPR converts .sil format files to .press for printing

FONTS (files containing characters of different styles)

TIMESROMAN*.AL several sizes of characters with serifs (like these)
 HELVETICA*.AL several sizes of characters without serifs (like these)
 MATH10.AL math and logic symbols for BRAVO

LOGO24.AL letters **XER** and **O**

HIPPO10.AL	Greek alphabet e.g. $\alpha\beta\xi\delta\Delta\tau\mu\dots$
GATES32.AL	arrows and things for use with SIL

This is a minimum set but it will get you started. Most of these files can be obtained easily by retrieving and executing the NPDISK command file. See the disk initialization procedure in the OPERATIONAL PROCEDURES chapter.

Other command files exist to simplify the retrieval of other file groups. These are the most commonly used; the list is by no means comprehensive.

BRAVO.CM	adds Bravo 7 (automatically retrieved by NPDISK.CM)
MESADISK.CM	makes an initialized disk into a MESA programmer's disk
PDISK.CM	makes an initialized disk into a BCPL programmer's disk
SIL.CM	adds Design Automation programs to an initialized disk

ACCOUNTS

If your machine is a part of the network you will probably need to obtain IVY and message accounts. IVY is a file storage facility (server) usually located at sites having ten or more Altos. (IVY's old name is IFS.) Stored on IVY are documents, files, and subsystems (computer programs or services) that you will need later on. Contact the local IVY administrator. In case you can't get an account right now, most IVY servers have a guest account, GUEST, password: Guest, that can be used to retrieve files. This is especially useful when retrieving files from other than the local IVY server.

If you are at PARC, you may need access to MAXC, a large computer that is part of the network. In general, MAXC accounts will not be issued to non-PARC people.

The question of message accounts is somewhat up in the air at the moment due to the changeover to Laurel. Once Laurel is in general use, local administrators will handle the assignment of accounts. Until then, message users must have access to a MAXC account. PARC, SDD, and ASD people are regularly assigned accounts. Group accounts are issued so that other individuals may also use the message facilities.

KEEPING UP-TO-DATE

The Alto exists in a rapidly changing environment. It requires some effort on your part to keep up-to-date, but you do have help. Some of the software developers use the message system to announce new or rereleased subsystems and packages, usually with updated documentation. The Whole Alto World Newsletter gathers these announcements together, along with unannounced changes, on a monthly basis. (It also includes documentation on newly released software, information about the hardware, notes on system operation, and papers on technologies affecting or affected by the Alto. If you would like to receive the Newsletter, contact the coordinator as indicated in the INTRODUCTION.)

Once you know of revised documentation, retrieve and print it. Using the Subsystems and Packages Manuals as a base, replace the updated sections with the documents just printed. Many of the documents contain a revision history as the last section. Review the changes indicated there and in the Newsletter and you will find that you can remain up-to-date with just a couple of hours effort each month.

IV. OPERATIONAL PROCEDURES

OPERATING THE ALTO

The Alto itself is a very simple machine for people to operate. There are only three things you need to know: loading a disk, booting, and turning it on and off. (Of course there are additional procedures for controlling each of the subsystems.)

ON/OFF *The Alto was designed and is intended to be left on at all times. When the machine is off, all lights are dark. Look for a red POWER light on the front of the machine. Older Altos (Alto I) generally have the power switch on the back of the display base; reach around to the right. On newer Altos (Alto II) the power switch is inside the processor cabinet (the big box on the floor). Using your thumbs, push in on the brushed aluminum plates (left and right front) and pull out a couple of inches. The switch is inside the top, left corner. Don't turn the machine off unless local procedures tell you to. **When turning ON or OFF always verify the the disk is off, i.e. the RUN/LOAD switch is in the LOAD position and the yellow READY light (right, front) is out and the white LOAD light is on.***

LOADING/UNLOADING A DISK The Diablo Model 31 disk drive used with the Alto has a protection mechanism that prevents the disk access door from being opened when the disk is spinning or when the power is off. If you look carefully through the smoked plastic panel, you can see a little flag that says LOCK just to the left of the disk pack hand hold when the disk is on (spinning). *Don't try to open the access door when this flag is up, something may break if forced.*

To load the disk, verify that the Alto is on, the LOAD/RUN switch (front, top, left) is in the LOAD position, and the white LOAD light is on. Open the access door by pulling out and down on the handle (cross bar at the top, front), and gently slide the disk in while holding the pack by the hand grip. Close the access door and push the LOAD/RUN switch to RUN. The white LOAD light will go out and, after about a minute, the yellow READY light will come on. The disk is now loaded.

To unload a disk, push the LOAD/RUN switch to LOAD, wait about about a minute for the white LOAD light to come on, open the access door, and slide the disk out. Some of the doors get sticky, so if it doesn't open easily, verify that the LOAD light is on and try again. (If you want to make really sure that it isn't locked, look for the flag through the smoked panel; it should be down.)

BOOTING The purpose of booting is to load into the Alto a copy of a program from an outside source. Normally, the Alto is booted from the disk (diskboot), so the first step is to load an initialized disk. If you have one, ready it. To boot using a disk, depress the boot button located on the back of the keyboard about an inch to the right of the thick, black cable. You should hear the disk rattle and a few seconds later, some text should appear at the top of the screen and a ">" about halfway down the left side. The boot operation is complete. The ">" is the prompt from the Executive; it is requesting you to type in a command. See the section below on USING THE EXECUTIVE.

If you don't have a disk, try to perform an ethernetboot. Depress and hold down the QUOTE and BACKSPACE keys on the keyboard, push and release the boot button while continuing to hold the other two keys until a small square with holes in it appears on the screen (several seconds). The keys can be released now. Some seconds after the square appears, it will disappear, some text will be displayed across the top of the screen, and ">" about halfway down the left side. The ">" is the prompt from the NetExecutive. This is not the same thing as booting off your own disk, only a few programs can be run this way. (Type "?" to list them). One of the things you may be able to do though is initialize a own disk over the network.

If the small square fails to appear, a boot server is not available; you will have to locate and ready an initialized disk to boot from.

CHECKING OUT YOUR MACHINE

There are several diagnostic programs that you can run to verify that the various pieces of your Alto are operating properly: DMT, CRTTEST, KEYTEST, and MADTEST. If the Alto is connected to the network, each can be obtained from a boot server, such as a Gateway. To execute any of them, boot over the Ethernet (boot while depressing the BS and quote keys) or type the netexec^{CR} command to the Executive. At this point the NetExec will appear on the screen. Type the name of the diagnostic to be performed and you're off and running. (Enter a "?" to list the boot files that can be called by the NetExec; the diagnostics listed above should appear. *Do not use DISKTEST. It's intended only for maintainers and could wipe out a readied disk.* Commonly used subsystems are also available.) If the tests indicate something is amiss, contact your local maintenance group.

THE WORKSTATION consists of the display, keyboard, keyset, and mouse. The action of the keys and mouse movement are tested by KEYTEST, the adjustment of the display by CRTTEST.

CRTTEST aids you in making a subjective judgement about the quality of the display's focus and linearity by drawing parallel vertical and horizontal lines. The thing to look at is the sharpness of the lines, especially near the edges of the display, and the shape of the boxes formed by the intersection of the lines. The boxes should be square, not tall or wide or diamond-shaped (romboid). There will be a little distortion at the corners so don't worry about that. The lines will be redrawn with a different spacing (three in all) when the space bar is depressed (or any other key for that matter). When finished either boot to get to the Executive or type SHIFT-SWAT to return to DMT.

KEYTEST will verify that each key makes positive contact, generating only one character. When the program starts, the keyboard, keyset, and mouse are drawn on the screen (the mouse may be hidden in the upper left hand corner). The display should picture the correct keyboard, either Alto I or Alto II. To flip to the other keyboard picture, move the cursor to the bottom of the screen (it should change to an arrow) and click any mouse button. Depress each key, one at a time; the corresponding key on the display should turn black. If it stays white, flickers, or more than one key turns black, there is a problem. When finished either

boot to get to the Executive or type SHIFT-SWAT to return to DMT.

THE DISK DRIVE There are no user diagnostics for the disk drive. Problems may be indicated by the need to run SCAVENGER on your disk more often than, say, once a month, though this may also indicate a bad disk. (The CopyDisk verify command, `v _erify DPO_ against DPOCR`, can be used to scan a disk for errors.) Regular preventative maintenance every three to six months should be sufficient to maintain proper head alignment and write current levels. *Do not run DISKTEST on a machine with a regular work disk readied; it may be destroyed by being overwritten.*

THE ALTO PROCESSOR can be roughly broken into several functional pieces: main memory, microprocessor memories (RAM and PROM), arithmetic-logic unit (ALU), registers, and data paths. It isn't necessary for you to know each of these or what they do, only that there are two diagnostics, DMT and MADTEST, which test the operation of most of the processor.

DMT tests the main memory, the area occupied by your data and most, if not all, of the subsystems that you run. When DMT is executed, the display will be black except for a small white square that bounces randomly about the screen.

DMT should be run for long periods, say overnight or over the weekend. If the Alto is left in the Executive, DMT will be called after 20 minutes. In the morning, when you come in and load the disk, while waiting for it to spin up to speed, depress and hold the 's' key. A three line message will be displayed a few inches from the bottom of the screen. The second and third lines should begin "0 Errors...". If some errors have been found, the memory chip location(s) will be indicated. *Don't boot your Alto; the machine may not work and the location information will be destroyed.* Inform the local maintenance group.

MADTEST, the Microcode Alto Diagnostic Test, exercises much of the rest of the processor. It is actually a collection of routines that test the RAM, PROM, ALU, registers, and data paths.

It is run in the same manner as KEYTEST and CRTTEST, namely, execute NetExec by booting from the Ethernet or typing `netexecCR` to the Executive, and then type `madtestCR`. The screen will indicate at the top the test routine being run, the number of passes completed, and the errors detected (if any). The middle of the screen will contain trash (the unformatted contents of memory) and below that a band containing the phrase "Black on white means DO TEST". Immediately below the band are the various tests that will be run.

The cursor consists of the usual arrow (like Bravo) with a changing series of black and white horizontal lines through it. The number and location of the lines change to let you know that something is happening because it isn't always apparent from the rest of the screen. The cursor also moves about the screen in its usual fashion except that sometimes it jumps (after a few seconds delay) rather than moving smoothly as is customary. You may also notice the screen "tearing". Both of these effects are a result of the level at which the diagnostics operate.

MADTEST will automatically run all the tests, over and over. The number of passes completed is displayed near the top of the screen; allow MADTEST to run several passes. If any errors are reported (they will be displayed about one-third the way down the screen), write them down and pass them along to the maintenance people. *Note: depressing the SPACE bar will halt the program and clear the screen. To suspend execution without clearing the screen, move the cursor into the area of the test list.* Moving it out will resume execution. After the test has run several passes, either boot to get to the Executive or type SHIFT-SWAT for DMT.

To prevent any of the tests from being run, move the cursor into the area of the test list and bug the test not to be run. It should change to white letters on a black background. To cause it to be executed, bug it again. Any tests shown in black letters on a white background at the start of a pass will be executed. The default is to execute all tests.

USING THE EXECUTIVE

This is the service that runs after a diskboot. It is used primarily for starting up other services, such as CopyDisk which can be used to initialize a disk.

STARTING A SERVICE To start a service, type its name followed by a RETURN (CR). For example

```
>copydiskCR
```

(If you did this, you will want to abort it. See ABORTING A SERVICE below.) Some services require additional information, say the name of a document on the disk. To display a document, "Notes", on the screen, use the service Type. Enter

```
>type notesCR
```

It doesn't matter if words are typed in capital letters, lower case, or a mixture of the two.

When typing a filename, it isn't necessary to enter the entire filename, only enough characters to unambiguously identify it. While entering a filename, if the ESC key is hit, the Executive will finish the name if it can. If the screen flashes black, there are two or more files that begin with the characters typed so far. Enter a few more characters (you needn't start over) and hit ESC again.

CORRECTING TYPING ERRORS Typing mistakes can be corrected using some special keys. To erase the last character typed use the BS (backspace) key. To erase the last word, type "w" while holding down the CTRL (control) key. Both the BS and w^C keys can be used to erase as many times as necessary. To erase the whole line and start over, type DEL (delete); it prints "XXX" and starts a fresh line.

ABORTING A SERVICE Usually a service can be suspended by *swatting*. To swat, depress the SWAT key while holding down the left SHIFT key. The location of the SWAT key depends on which style of keyboard you have. On the Alto II keyboard there is a vertical column of five keys to the right of the standard keys; the SWAT key is the top key on the right column. On

the Alto I, the SWAT key is the blank key in the lower right corner. In both cases it is blank.

After swatting, the service SWAT takes over. To get back to the Executive type k^C (depress the "k" key while holding down the CTRL key); to resume the service that was running, type p^C (this doesn't work with some services).

INITIALIZING/INITIALIZING A NEW DISK

There are several ways to initialize a new disk, some better than others for reasons that will be explained. *Usually local procedures will describe the best method for obtaining and building new disks at that site.* One method, using the network facilities (ethernet and file server) requires that your machine be a part of the network and that you have access to a file server (IVY or MAXC). An alternate method, copying an existing disk, requires an initialized disk and either an Alto with two disk drives or two Altos connected by an ethernet.

In general, initializing a disk over the ethernet is preferred. Unless the source disk used for copying has been specifically built for that purpose (and not used otherwise) the new disk will contain an unknown assortment of programs and fragmented freespace.

COPYING - DUAL DRIVE ALTO You've located an initialized disk and a dual drive Alto. Load the initialized disk into the lower drive (called DP0) and the new disk into the upper drive (DP1). After both are ready, boot the machine (ethernetboot preferred) and perform the following dialog.

>copydisk^{CR} (The screen will change appearance)

NOTE: CopyDisks elsewhere on the net can't write on your disks. Type "WRITEPROTECT" to allow it.

*writeprotect off

*copy from dp0^{CR} (Zero, not Oh)

Copy to dp1^{CR}

Copying onto dp0 will destroy its old contents.

Are you sure this is what you want to do? [Confirm] y _es

Are you still sure? [Confirm] y _es

The machine will now copy the lower disk, DP0, to the upper disk, DP1, verify that the two disks are identical and then ask for another command. Type

*quit^{CR}

After installing (see INSTALLING A DISK), the new disk can be used to boot the Alto and to store programs and data.

COPYING - TWO ALTOS You've located an initialized disk and two Altos on the same ethernet. Etherboot each machine. If this is successful, load each machine with a disk.

If the Altos won't boot without a disk, load the initialized disk into one of the Altos, boot it (boot button only), and type `copydiskCR`. Without disturbing the keyboard, unload the that disk and load the new disk. Now take the good disk to the other Alto, load the disk, boot, and type `copydiskCR`.

Both machines should now be running the copydisk program and their screens should look the same. At the Alto containing the new disk, perform the following dialog.

NOTE: CopyDisks elsewhere on the net can't write on your disks. Type "WRITEPROTECT" to allow it.

`*writeprotect _off`

`*copy from [nnn#]dp0CR` (where nnn is the number of the Alto with the good disk. Look in the black band.)

`copy to dp0CR`

Copying onto dp0 will destroy its old contents.

Are you sure this is what you want to do? [Confirm] y _es

Are you still sure? [Confirm] y _es

The machines will now copy the initialized disk to the new disk, verify that they are the same, and ask for another command. Type `quitCR` at both machines. The new disk should now be installed (see INSTALLING A DISK) and then can be used to store programs and data files such as documents.

USING THE NETWORK You will need a new disk, an Alto connected to a network with a file server (IVY or MAXC), and access (an account) to the server. Again, local procedures should be consulted if available.

Load the new disk, perform an etherboot, type `operating-systemCR`, and answer the questions as indicated.

Do you want to install this operating system? y _es

Do you want the long installation dialog? y _es

Do you want to ERASE this disk before installing? y _es

Type the name of a host from which I can get Alto programs: name of file server^{CR}

Type the name of the directory where Alto Programs are kept: Alto^{CR} (Usually)

If you wish to change disks, please do so now. When the disk is ready

type OK to proceed, A to abort: OK^{CR}

The disk is configured with the multiple-version feature enabled.

Do you want to change this setting? n _o

Do you want to change the error logging address (currently x#yyy#zz#)? n _o

Do you want to disable parity error detection? n _o

Do you want to disable phantom parity reporting? n _o

What is your name? Ludolph^{CR} (Generally identical to your IVY/MAXC account)

Please give your disk a name? Whole Aito World^{CR}

Do you wish to give the disk a password y _es or n _o

What is the password? asdfgh^{CR} (Generally identical to your IVY/MAXC account)

The system will now boot itself and call the FTP service. This service retrieves files from file servers, in this case the one you named in the preceding dialog. You will need a valid account on that server. The Executive and FTP programs will be retrieved in order to assure your having the latest versions on your disk. Disk installation is a part of this procedure, so it won't have to be done again. The disk is pretty empty so you will want to perform the operations under **RETRIEVING A BASIC SET OF FILES** below.

INSTALLING A DISK Disk installation (short dialog) is used to identify the owner of a disk, to give the disk a name, and to optionally require a password each time the disk is booted. To install a disk, load it, boot the Alto, and type `INSTALLCR`. The system will then start a dialog with you.

Do you want the long installation dialog? n_o
 What is your name: Ludolph^{CR} (Generally identical to your IVY/MAXC account)
 Please give your disk a name: Whole Alto World^{CR}
 Do you wish to give the disk a password? y_es or n_o
 What is the password: asdfgh^{CR} (Generally identical to your IVY/MAXC account)

The system will now boot itself, request the password if specified, display the owner's and disk's names at the top of the screen, and prompt for a command.

RETRIEVING A BASIC SET OF FILES Now that you have a very basic disk you will want to assure that it contains the subsystems required for its intended use. If it was created using DiskCopy, it probably has what you need. Typing a `TAB` will list the files that are on the disk. Check the list in **WHAT YOU NEED** against the names displayed on the screen. If most of the files aren't there or if the disk was initialized over the Ethernet, you will probably want to use FTP to retrieve and execute a command file which will, in turn, retrieve the files you need. To do this type the following line to the Executive where "file server" is the name of the file server you have access to:

`>FTP file server RET/C <ALTO>command fileCR` (Retrieves the command file)
`>@command file@CR` (Executes the command file)

where "command file" is `NEWNPDISK.CM` if the disk is for document creation or `NEWPDISK.CM` if it is for BCPL programming.

The FTP program will open a connection to the file server and retrieve the command file. That file contains commands that in turn will use FTP to retrieve a number of files. When everything finally comes to a stop you will probably be left in Bravo. Type a `qCR` to quit and return to the Executive. Your disk is now ready to do some work.

NORMAL DISK MAINTENANCE

The Executive provides routines to list filenames, list a file's attributes (e.g. type, size, and creation date), display a file's contents, delete a file or files, and group files together as a single file for storage or shipment (and the inverse operation). Two programs, `Put` and `DDS`,

can be used to simplify file deletion. Copying files between drives on a dual disk Alto can be performed by Put and Copydisk. The copying of files to and from the Alto disk is done using FTP.

LISTING FILENAMES To find out if a particular file is on the Alto disk, type the name of the file followed by a TAB. All the files whose names begin with the typed-in characters will be displayed on the screen. If, while entering a command to the Executive, you get part way through the filename but don't remember how it ends, type a "?". All the filenames beginning with the characters typed so far will be listed on the screen. Find the name you need and finish typing the name.

If you name your files in a systematic way, groups of filenames can be listed by substituting *pattern characters* for parts of the name, followed by a TAB. The two pattern characters are "#" and "*". Any single character can replace a "#"; any string of characters can replace a "*". For example "*.memo" represents any filename ending in ".memo" while "#.memo" represents only the files that have three character main names followed by the extension ".memo", such as "ABC.memo". Any combination of pattern and filename characters is valid.

LISTING FILE ATTRIBUTES Enter the Executive command, File filename, to list a files size, type, creation and write dates, and serial number and disk address.

DISPLAY A FILE'S CONTENTS Enter the Executive command, Type filename. A portion of the file will be displayed, followed by the word MORE?. Type n to terminate, SPACE to procede.

DELETING FILES To delete a file, type delete filename1 filename2 ...^{CR}. Pattern characters can be used in the filenames, but *be careful because once deleted, you can't get a file back*. If there is more than one version, only the oldest version (the one with the lowest number) is deleted.

GROUPING FILES Many related files may be grouped as a single unit for storage using the Executive command, Dump dumpfilename filename filename... The dumpfilename usually has a .dm extension regardless of content. To recover the individual files, enter Load dumpfilename.

Additional details on the preceeding operations is contained in the Executive section of the Subsystems Manual. For information on the capabilities and use of the other programs mentioned in the first paragraph, see their respective subsections in the Subsystems Manual (except for Put, which has self-contained documentation).

DISK CRASH!

There are various ways the Alto disk can be damaged. The most typical symptom is the failure to boot properly. Using Scavenger and various techniques, it is almost always possible to recover a disk or, at the minimum, the irreplaceable files on it. Since a fair amount has already been written, it won't be repeated here. The discussions can be found in:

Alto User's Handbook	pp. 8-9
<AltoDocs>OS.press	pp. 30-33
<AltoDocs>Subsystems.press	Scavenger section

All of this discussion has been brought together, along with additional comment on [IFS-2]KSDSupport>SDDocs>Scavenger.bravo.

THE USER COMMAND FILE

This file contains user-settable default information for many of the system's services. It should be on the disk now, but if not try retrieving New-User.cm from the <Alto> directory on your local file server and rename it (rename oldname newname). Some of the settings may have to be altered before placing the disk in general use. Of specific interest are the [HARDCOPY] and [EXECUTIVE] entries. To see what they contain, enter the command type user.cm^{CR} to the Executive. This will display on the screen some of the text in the file User.cm. When you have finished looking at what is on the screen, touch the SPACE bar (actually, any key except "n" will do). When the whole file has been displayed, Type will return to the Executive. If you want to abort Type, enter "n" when it asks "More?".

Look at the [EXECUTIVE] entry; it's usually the first one. You should see something like the following. Ordering isn't particularly important.

```
[EXECUTIVE]
eventBooted: // eventBooted
eventRFC: FTP// eventRFC
eventInstall: // eventInstall
EventAboutToDie: // eventAboutToDie
eventUnknown: // eventUnknown
eventClockWrong: Settime // eventClockWrong
```

In this case, the words on the left are things (events) that happen that the Executive knows about. If one of these events occur, the Executive will call the service listed following the ":". For example, if the Executive notices that the time value it has doesn't look like a valid time value, it will call the service Settime which will attempt to get the time from another source on the network or, failing that, ask you for the correct date and time. The "/" is the start of a comment that is displayed on the screen when that event occurs. Many people change "eventBooted" to say "Hi (your name)", then, everytime the disk is booted, the screen displays a greeting.

If you have a message account you may wish to call MailCheck when you boot, like so:

```
eventBooted: Mailcheck// Hi Frank
```

To change the User.cm you will want to use Bravo or one of the other text editors. Because they take a little experience to learn, altering the User.cm won't be described here. You should look at the chapter, WHAT YOU NEED, to see about getting a text editor and its documentation.

The other entry you should look at is [HARDCOPY]. This one might actually have to be changed so make a note to do it later when you learn how. This entry defaults where documents on your disk are sent for printing. This assumes that your machine is part of a network that has an attached printer. If so you will need to learn its name before making the change. The entry looks like this:

```
[HARDCOPY]
PREFERREDFORMAT: Press
EARS:
PRESS: Menlo
PRINTEDBY: "your name"
```

There are two formats used to encode documents for printing, Press and Ears. The one you indicate as preferred should be the one you intend to use most often. Since there is only one Ears printer and it is located in Palo Alto, you probably want to specify "Press". The "EARS" and "PRESS" entries identify the default printers you would like your documents to go to. Even if you specify "Press" as your preferred format, if you are in Palo Alto you may want to use the Ears printer from time to time so that entry should read

```
EARS: Palo
```

If you're not in Palo Alto, you needn't bother. You will want to specify the name of your local press printer, usually either a Dover or Sequoia. If you know what ethernet your on, look at the Alto Network diagram for the name.

As an alternative to specifying the printer's name, you may wish to specify its address in the form nn#mmm#. nn is the ethernet's address; mmm is the address of the Alto that drives the printer. The reason for doing this is to cover the infrequent occasion when the name server, usually a gateway, is down. These numbers can also be found on the Alto Network diagram. *Don't forget to reinstall Bravo (Bravo/i) after editing this section.*

There are also entries for other services; Bravo, DDS, Chat, SIL, and Draw are commonly used. See the documentation on each of these for descriptions of their entries.

PRINTING

Almost all documents are stored in electronic form on file servers. As a result, you will have to retrieve and print the documents you need. Printers that run Spruce and Press (Spruce is run on Dover and Sequoia) software require that files sent to them for printing be in press format. However, several other formats (bravo, gypsy, tty, text, and some ears) can be converted and printed as described below.

You should first assure that the [HARDCOPY] section of the User.cm file is in the proper format, that is, it should reference a nearby printer. While each of the printing services provides override facilities, it is easier and less error prone if you start with a HARDCOPY specification. See the USER COMMAND FILE section for a full explanation.

PRESS FILES Empress will send a press file to the press printer indicated in the HARDWARE section of your User.cm file; just type **Empress filename**. The number of copies can be specified as well as a different printer. See the Empress section of the Subsystems manual for a complete description.

BRAVO FILES Bravo has a Hardcopy command that will properly format and transmit the current workfile to the printer specified in the [HARDCOPY] section of the User.cm file. Options are available to specify a different printer (@printer name^{ESC}) and multiple copies (cnumber of copies^{ESC}). An option also exists to create a press file without printing.

EARS FILES *Some* ears files, which are normally printed only on the ears printer at PARC, can be converted to press format for printing on a local printer. The ears file must contain a defined font set and may not contain diagrams. The only way to discover this is to try to convert it by calling Pressedit, **Pressedit Name.press ← Name.ears**. If successful, this will create a press format file on the Alto disk which may be treated as any other press file. See the Pressedit section in the Subsystems Catalog for more detail.

GYPSY FILES Gypsy, like Bravo, will format its workfile into press format and transmit it to the printer specified in the User.cm file.

TEXT FILES The various text editors, e.g. Bravo, Gypsy, and UGH, maintain their source as a text file. Empress will automatically recognize and convert text files to press format before transmitting them to the printer. However, this is not commonly done. Empress does not attempt to use the formatting information in these files; it is simply stripped off. The Bravo and Gypsy editors both contain facilities to properly format their workfiles in press format and transmit them to the printer.

TTY FILES Empress will automatically recognize and convert .tty files to press format before transmitting them to the printer: just type **Empress filename**. The same options that apply to press files apply to tty files. See the Empress section of the Subsystems manual for a complete description.

OTHER PRINTERS Dovers and Sequoias are not the only printers, though they are the most common. Other printers include the Versatec electrostatic printer, the Diablo HyType, and the Slot 3100. The Versatec and Slot 3100 normally run press software which will print press files as described above. If they have been set up as servers, Empress can be used to transmit the files to the printer, otherwise the file to be printed will have to be transferred between machines using FTP.

The HyType was the original hardcopy printer used with the Alto and is available as an option. Both Bravo and Gypsy will output their workfiles to this typewriter like device; for Bravo, use the Hardcopy command with the D option. Though the output is right justified, it is printed in a single font.

ShowAIS Documentation

April 4, 1978

ShowAIS is a program for display 8 bit per sample AIS files on the Alto display. Two versions are available: [IVY]<Maleson>ShowAIS.RUN requires the AIS file to be on local disk storage, and [IVY]<Maleson>NetShow.RUN displays AIS files from a remote server site. The network version is available as ShowAIS.RUN from all gateway boot servers.

ShowAIS will display any rectangular window of the AIS file. The window specification commands are:

- XS (XStart) -- first pixel to display in the x direction (default=0)
- YS (YStart) -- first pixel to display in the y direction (default=0)
- XL (XLength) -- number of pixels to display in x (default=x length of image)
- YL (YLength) -- number of pixels to display in y (default=y length of image)
- L (Length of scan) -- number of horizontal output dots to paint on the Alto screen (default=608)
- B (Black) -- all pixels less than or equal to this value will print as black.
- W (White) -- all pixels greater than or equal to this value will print as white.
- R (Reset) -- restore window settings to default values.
- S (Show Settings) -- display current settings.

To display the currently selected window on the screen, the command is:

- P (Print Picture) -- the current window is displayed on the screen

Additional useful commands are:

- E (Erase Screen)
- I (Invert Display) -- turn all black dots white, and all white dots black.
- <ESC> -- Remove (or Restore) the six line text display from the screen.
- D -- Display on or off: the speed of printing a picture is approximately doubled when the display is turned off (the area painted so far is indicated by a black rectangle of increasing size).
- Q (Quit) -- In the network version, a new AIS file name is requested (hitting carriage return will terminate the program). In the disk version, Q immediately terminates the program.

In addition to the manual commands for setting the window, a window may be specified interactively using the mouse. After typing the appropriate command, point to one corner of the desired rectangle; hold down any mouse button as you move the mouse to the opposite corner of the desired rectangle. A flashing area will indicate the window being selected. Releasing the mouse button causes the new window to be displayed. The commands are:

- U (Update zoom) -- display the indicated rectangle, and update the current window parameters (XS,YS,XL,YL) to be the rectangle coordinates.
- Z (Zoom) -- display the indicated rectangle, but leave the current settings active. (After a zoom, the previously displayed image can be restored by typing P).

ALTOII MEMORY LAYOUT

BIT	0-17777		20000-37777		40000-57777		60000-77777		100000-117777		120000-137777		140000-157777		160000-176777	
	EVEN	ODD	EVEN	ODD	EVEN	ODD	EVEN	ODD	EVEN	ODD	EVEN	ODD	EVEN	ODD	EVEN	ODD
0	1-16	1-18	1-26	1-28	1-36	1-38	1-46	1-48	1-56	1-58	1-66	1-68	1-76	1-78	1-86	1-88
1	2-16	2-18	2-26	2-28	2-36	2-38	2-46	2-48	2-56	2-58	2-66	2-68	2-76	2-78	2-86	2-88
2	3-16	3-18	3-26	3-28	3-36	3-38	3-46	3-48	3-56	3-58	3-66	3-68	3-76	3-78	3-86	3-88
3	4-16	4-18	4-26	4-28	4-36	4-38	4-46	4-48	4-56	4-58	4-66	4-68	4-76	4-78	4-86	4-88
4	1-11	1-13	1-21	1-23	1-31	1-33	1-41	1-43	1-51	1-53	1-61	1-63	1-71	1-73	1-81	1-83
5	2-11	2-13	2-21	2-23	2-31	2-33	2-41	2-43	2-51	2-53	2-61	2-63	2-71	2-73	2-81	2-83
6	3-11	3-13	3-21	3-23	3-31	3-33	3-41	3-43	3-51	3-53	3-61	3-63	3-71	3-73	3-81	3-83
7	4-11	4-13	4-21	4-23	4-31	4-33	4-41	4-43	4-51	4-53	4-61	4-63	4-71	4-73	4-81	4-83
8	1-17	1-19	1-27	1-29	1-37	1-39	1-47	1-49	1-57	1-59	1-67	1-69	1-77	1-79	1-87	1-89
9	2-17	2-19	2-27	2-29	2-37	2-39	2-47	2-49	2-57	2-59	2-67	2-69	2-77	2-79	2-87	2-89
10	3-17	3-19	3-27	3-29	3-37	3-39	3-47	3-49	3-57	3-59	3-67	3-69	3-77	3-79	3-87	3-89
11	4-17	4-19	4-27	4-29	4-37	4-39	4-47	4-49	4-57	4-59	4-67	4-69	4-77	4-79	4-87	4-89
12	1-12	1-14	1-22	1-24	1-32	1-34	1-42	1-44	1-52	1-54	1-62	1-64	1-72	1-74	1-82	1-84
13	2-12	2-14	2-22	2-24	2-32	2-34	2-42	2-44	2-52	2-54	2-62	2-64	2-72	2-74	2-82	2-84
14	3-12	3-14	3-22	3-24	3-32	3-34	3-42	3-44	3-52	3-54	3-62	3-64	3-72	3-74	3-82	3-84
15	4-12	4-14	4-22	4-24	4-32	4-34	4-42	4-44	4-52	4-54	4-62	4-64	4-72	4-74	4-82	4-84
HO	1-20		1-30		1-40		1-50		1-60		1-70		1-80		1-90	
H1	2-20		2-30		2-40		2-50		2-60		2-70		2-80		2-90	
H2	3-20		3-30		3-40		3-50		3-60		3-70		3-80		3-90	
H3	4-20		4-30		4-40		4-50		4-60		4-70		4-80		4-90	
H4	1-15		1-25		1-35		1-45		1-55		1-65		1-75		1-85	
H5	2-15		2-25		2-35		2-45		2-55		2-65		2-75		2-85	
P	3-15		3-25		3-35		3-45		3-55		3-65		3-75		3-85	

NOTE: 1. Card-Chip location.
2. The Memory Configuration Switch complements the high-order memory address bit

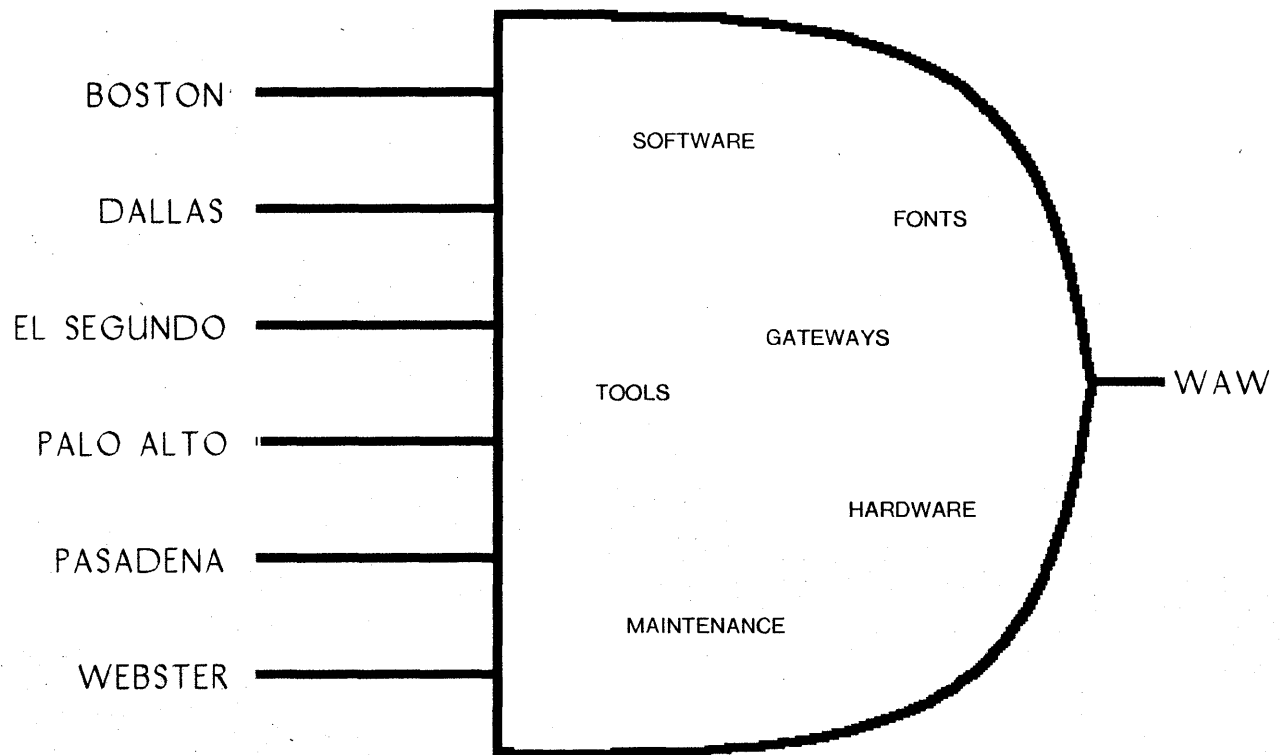
SSL ALTO II
EXTENDED MEMORY LAYOUT

BIT	BANK 0				BANK 1				BANK 2				BANK 3			
	00000-77777		100000-177777		00000-77777		100000-177777		00000-77777		100000-177777		00000-77777		100000-177777	
	EVEN	ODD	EVEN	ODD	EVEN	ODD	EVEN	ODD	EVEN	ODD	EVEN	ODD	EVEN	ODD	EVEN	ODD
0	1-16	1-18	1-26	1-28	1-36	1-38	1-46	1-48	1-56	1-58	1-66	1-68	1-76	1-78	1-86	1-88
1	2-16	2-18	2-26	2-28	2-36	2-38	2-46	2-48	2-56	2-58	2-66	2-68	2-76	2-78	2-86	2-88
2	3-16	3-18	3-26	3-28	3-36	3-38	3-46	3-48	3-56	3-58	3-66	3-68	3-76	3-78	3-86	3-88
3	4-16	4-18	4-26	4-28	4-36	4-38	4-46	4-48	4-56	4-58	4-66	4-68	4-76	4-78	4-86	4-88
4	1-11	1-13	1-21	1-23	1-31	1-33	1-41	1-43	1-51	1-53	1-61	1-63	1-71	1-73	1-81	1-83
5	2-11	2-13	2-21	2-23	2-31	2-33	2-41	2-43	2-51	2-53	2-61	2-63	2-71	2-73	2-81	2-83
6	3-11	3-13	3-21	3-23	3-31	3-33	3-41	3-43	3-51	3-53	3-61	3-63	3-71	3-73	3-81	3-83
7	4-11	4-13	4-21	4-23	4-31	4-33	4-41	4-43	4-51	4-53	4-61	4-63	4-71	4-73	4-81	4-83
8	1-17	1-19	1-27	1-29	1-37	1-39	1-47	1-49	1-57	1-59	1-67	1-69	1-77	1-79	1-87	1-89
9	2-17	2-19	2-27	2-29	2-37	2-39	2-47	2-49	2-57	2-59	2-67	2-69	2-77	2-79	2-87	2-89
10	3-17	3-19	3-27	3-29	3-37	3-39	3-47	3-49	3-57	3-59	3-67	3-69	3-77	3-79	3-87	3-89
11	4-17	4-19	4-27	4-29	4-37	4-39	4-47	4-49	4-57	4-59	4-67	4-69	4-77	4-79	4-87	4-89
12	1-12	1-14	1-22	1-24	1-32	1-34	1-42	1-44	1-52	1-54	1-62	1-64	1-72	1-74	1-82	1-84
13	2-12	2-14	2-22	2-24	2-32	2-34	2-42	2-44	2-52	2-54	2-62	2-64	2-72	2-74	2-82	2-84
14	3-12	3-14	3-22	3-24	3-32	3-34	3-42	3-44	3-52	3-54	3-62	3-64	3-72	3-74	3-82	3-84
15	4-12	4-14	4-22	4-24	4-32	4-34	4-42	4-44	4-52	4-54	4-62	4-64	4-72	4-74	4-82	4-84
	BANK 0				BANK 1				BANK 2				BANK 3			
H0	1-20		1-30		1-40		1-50		1-60		1-70		1-80		1-90	
H1	2-20		2-30		2-40		2-50		2-60		2-70		2-80		2-90	
H2	3-20		3-30		3-40		3-50		3-60		3-70		3-80		3-90	
H3	4-20		4-30		4-40		4-50		4-60		4-70		4-80		4-90	
H4	1-15		1-25		1-35		1-45		1-55		1-65		1-75		1-85	
H5	2-15		2-25		2-35		2-45		2-55		2-65		2-75		2-85	
P	3-15		3-25		3-35		3-45		3-55		3-65		3-75		3-85	

NOTE: 1. Card - Chip location.
2. The Memory Configuration Switch reverses high and low memory within a bank.

WHOLE ALTO WORLD MEETING

JUNE 1, 1978



KEEP IN SYNC

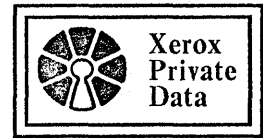
9:00am to 3:30pm

Executive Dining Room

El Segundo, Ca

Whole ALTO World Newsletter

Technology and Tools



XEROX

MAY 31, 1978

SPECIAL NOTES

FONT CHANGE COMMING - As detailed in the attached paper on printing, PARC will be changing its printing fonts on June 15th. Similar changes are expected to occur at other sites at about the same time. This will require that everyone retrieve a new Fonts.widths file on each of their disks soon after local Spruce installers announce the change at their sites. Users with old Fonts.widths will find their output rather unattractive as a result of bad spacing and ragged edges.

GENERAL NOTES

WHOLE ALTO WORLD MEETING - The Whole Alto World meeting will be held on June 1 at the Cockatoo Inn near the El Segundo facility. Our host is Dick Sonderegger, SDD. See next month's Newsletter for an account of the happenings.

TOOLS

HARDWARE

ALTO BUILD - An 8th build will take place during the last quarter of this year. Requirements should have been made known to Terry Haney, SPG, by June 1. The configuration is being altered slightly by replacing the current keyboard with an Alto I form/function compatible keyboard and deleting the five-finger keyset. Transport trays, a rolling platform for the Alto which still permits it to be placed under a table, may be ordered as an option.

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local IVY server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [XEOS].

NEW RELEASE: GYPSY - This subsystem has been significantly enhanced so it is being reported here rather than under ReReleases. GYPSY is a text editor originally derived from an early version of Bravo. It is modeless, that is, you don't have to enter "i" before inserting text, followed by ESC when finished; just position the cursor and type. It has only simple formatting facilities, so it is more suited to programming than document creation. It also provides a hierarchical file facility to group documents, i.e. files, into "folders".

One of the enhancements enables GYPSY to replace Executive.run. Replacing Bravo and Executive with GYPSY will free over 200 disk pages, reduce typing, and integrate source code editing and compilation activities. After booting, the screen will contain several *Find* and *Execute* menu items (see the attached hardcopy of the GYPSY screen). Bugging an *Execute*



item will cause the subsystem in the associated {...} to be run. Bugging a *Find* item will cause that folder to be retrieved and a new screen will be displayed containing a *Fetch* menu item for each file. Bugging *Fetch* will open the associated file for editing. These menu lists can be expanded by the user at will.

GYPsy permits text within the file to be "executed"; simply select the text and bug *Do It* (in the window above the text window). For example, if the text selected is a compile command, contained as a comment in the source file, the compiler will be called after automatically updating the changes. Following compilation, control is returned to GYPsy. Bugging *Find* and *Fetch* opens the file for additional editing as necessary. A second text window can be opened so that both the source and error files can be viewed simultaneously.

A "hardcopy" facility will immediately format and transmit the current file to a printer, or Empress can be used to print a GYPsy file. GYPsy formatted hardcopy is in a double column format.

Due to its bulk, the documentation is not reproduced here. Retrieve <AltoDocs>GYPsy.press.

NEW RELEASE: HARDCOPY - This subsystem, designed by Jay Israel to save you keystrokes and concentration, will automatically invoke the Bravo "hardcopy" command from the Alto EXECUTIVE. It will also call FTP if necessary to retrieve files from a shared file server such as MAXC or IVY. Retrieve <Alto>Hardcopy.run. The documentation is appended to the Newsletter.

NEW RELEASE: SOFTBITBLT - This package, consisting of three files by Dave Boggs, emulates the BitBlit instruction in software. It can be retrieved by loading <Alto>SoftBitBlt.dm. The documentation is attached to the Newsletter.

ReReleases - Subsystems

EMPRESS - The changes are of a minor, internal nature. The documentation is unchanged.

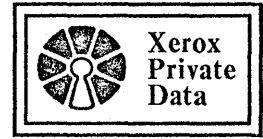
GOBBLE - A part of the SIL system, Gobble has been modified to treat F-ICType identically to N-ICType so that the same Dict.Analyse can be used for GOBBLE and ROUTE. Retrieve <SIL>Gobble.run. The documentation is unchanged.

GYPsy - Extensive enhancements have been made which enables press output of hardcopy and permits GYPsy's use in lieu of Executive.run. See the GYPsy entry under NEW RELEASE for a more detailed discussion. Load <Alto>GYPsy.dm and the new documentation, <AltoDocs>GYPsy.press.

MENUEdit - A part of the BCPL MENU package, this releases corrects isolated cases in which DCBs were incorrectly written. See the MENU package below.

MICROD - The nature of the changes are unknown to me. Retrieve <Alto>Microd.run. The documentation is unchanged.

READPRESS - The new version fixes a bug in the interpretation of the SkipControlBytes command. Retrieve <Alto>ReadPress.run. The documentation is unchanged.



ReReleases - Packages

MENU - This version, 1.2, uses BitBlit to manipulate the screen, permits the use of a colon in a string, generates default names if required, and no longer generates zero DCBs. It is compatible with the previous version. Load <Alto>Menu.dm. The documentation is unchanged.

TECHNOLOGY

This month's paper is on printing within the Xerox Alto community. Dan Swinehart, Joe Maleson, Patrick Baudelaire, and Edward Fiala have put together an overview of existing hardware and software with an extensive list of references and a glossary. Although it has many references to PARC specific items, it is a basic document whose breadth should make it of interest to all who use the Alto.

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC]<AltoDocs>WAWnews.press or may be obtained from the editor, Frank Ludolph, XEOS, by messaging <Ludolph> or calling Intelnet 8*923-4356.

XEROX

PALO ALTO RESEARCH CENTER

Computer Science Laboratory

May 11, 1978

To PARC/SDD
From Dan Swinehart, Joe Maleson, Patrick Baudelaire, and Edward Fiala
Subject Printing at Palo Alto
Filed As [Maxc]<Gr-Docs>Printing.Press

This document has the twofold aim of describing the printing facilities that are currently available, and of announcing two major impending changes.

Key Points

We are currently in transition from one major printing format (EARS) to another (Press), with a corresponding introduction of new printing hardware (*Dover*, *Pimlico*, *Sequoia*). Some aspects of this transition will affect users quite a bit, some very little. All will of course affect those who produce formatted files.

One upcoming change that will have substantial effect on the entire community will be the distribution, on June 15, of a new Fonts.Widths file, correcting the width specifications for our major document fonts. Previously produced Press files should continue to print reasonably accurately, although they may not do so forever. Subsystems that create Press files will be required to supply the current date (machine readable format) in a new field within each new Press file.

Another major impact will be felt as the *Ears* system is decommissioned on the same date (June 15, 1978). It will be possible to convert most EARS format files to Press format, and thus to print them. This includes all new EARS files that are produced by the *Pub* system, and most if not all previously created ones.

A modification of printing resolution (from 350 to 384 dots/inch) for *Dover* and *Sequoia* printers will limit printing to the lower 10.6" of each page; information in the top .4" will be lost. This is expected to affect very few documents.

The sections that follow survey the current state of *Printing Formats and Transmission Protocols*, *Fonts*, *Printing and Formatting Programs*, *Printer Types*, and *Impending Changes*. Appendices supply a variety of useful information, including a comprehensive glossary of terms associated with page imaging. There is also a bibliography of related documents and manuals. Feel free to skim or skip those sections that do not appear to be of interest.

Printing Formats

EARS Format

EARS format [Rider1] was developed as the input format for the *Ears* printer (see below). It is a character oriented format, specifying for each page the names and positions of each character to be printed. An EARS format file must also contain a font directory before it can be printed. This directory, describing the shape of each character to be printed, can be a permanent part of the file, or may be inserted just before the file is shipped to the printer, based on a list of requests (the *Doculist*) that is embedded in the document. Exception: a default set of 16 commonly used fonts is available at the printer site, and need not be transmitted with the file.

In addition to characters, line drawings and passable low-resolution bit maps may be specified using EARS format, by creating suitable "type faces" containing the required patterns. *Ears* can handle most Alto screen resolution bit maps in this way.

In Palo Alto, there is only one *Ears* system (named *Palo*).

Press Format

Press format [Newman/Sproull] is 'an attempt to define a "universal" format for describing documents' for printing and editing. Its major goal is the complete description of document appearance in a device-independent fashion. A Press file should look the same, within device limitations, on any printer.

Press format allows the specification of:

- Text characters.

- Solid rectangles parallel to the paper edges.

- Objects whose outlines are defined by a series of graphic commands including spline curves. Character shapes may be defined in this way. (see *fonts* section below).

- Images specified as a matrix of intensity values at arbitrary resolutions. Bit map patterns are an instance of such images, where intensity values are constrained to one bit.

Any of these entities may be placed at any location on the page, and may be assigned brightness (obtained with half-tone shading methods) and color characteristics. Characters to be printed are specified by name. Custom shapes may be included by sending their splines and the names to be assigned to them, but one usually depends on the server to locate the necessary shapes based on the names alone, using established conventions (see the *Fonts* section below).

A printer that accepts Press format is not required to implement all its provisions. The descriptions below indicate for each program and printer what subset of the Press facilities it implements.

Text Files

Programs exist on both Maxc and Altos (see below) for converting standard Ascii text files to EARS or Press format, with optional headings and page numbers. These programs often combine translation with transportation to the appropriate printer. For the results to be pleasing, the text files must contain explicit line breaks.

Pspool Format

Pspool format [Fiala1] is a simple augmentation of Ascii text files. Escape sequences describe margin settings, column settings, landscape/portrait selection, font selection, underlining, etc. A document heading may override default selections for user identification, file identification, etc.

At present, only the printing subsystem on *Maxc1* or *Maxc2* (see *Formatting Programs*, below) will honor Pspool format, and then only for files to be sent to *Ears*. A Press version should be complete by June 15.

Transmission Protocols

A printer is classified as a "server" if it can operate unattended, accepting files sent to it over the internetwork using a *Pup*-based protocol. Otherwise it is a "stand-alone" printer, requiring each user to attend its operation.

The user of a stand-alone printer may choose any available transmission program to transfer files to the printer's local disk. The *Ftp* program [Boggs/Taft], using the FTP protocol [Shoch1], is preferred because of its speed, utility, and availability -- it is the only protocol currently used by the *IFS* systems.

All server systems currently use a simpler protocol, EFTP protocol [Shoch2], which is more easily incorporated into sending programs like *Bravo* that are tight for space. The EFTP protocol provides only for the transmission of file data. Thus the only identification information available to the server is information contained in the file itself. This, combined with some limitations in Press format, has caused some difficulty in properly identifying the sender of documents to Press printers.

A final protocol, known as EARS protocol [Taft2], is honored by all servers. It provides a means whereby user programs can determine a server's current status.

See the *Formatting Programs* section for more information about protocol usage.

Fonts

High resolution raster representations of fonts comprise many bits, and are device dependent; for these reasons font rasters are not included in Press files. Instead, fonts are specified by family name, a numeric code specifying type face (bold, italic, regular, etc.), point size, and rotation. An extensive description of font issues can be found in [Sproull4]. Two representations exist for fonts: outlines (described by graphic commands including spline curves) and rasters (or "bit maps"). Along with the description of character shape according to one of these representations, positional information is needed. A variety of file formats are presently used for storing this font information. Sometimes several fonts are included in a single file, called a dictionary. *PrePress* [Sproull2] is a program which converts among the various formats, and performs some bookkeeping operations for font dictionaries. Press format allows up to 16 different fonts per *entity* (a page subgrouping). When more than 16 fonts appear in a page, they must be grouped into *font sets* (of up to 16 fonts) in such a way that each page contains fonts from only one set. A document may contain up to 64 font sets.

Font Naming Conventions

A standard file naming convention allows identification of what font is contained in a file. The convention is:

{family-name-in-full}{point-size}{[B|L] }{[I] }{[C|E] }.{extension}

Examples: TimesRomanB.SD, TimesRoman10B.AC

The optional B|L stands for Bold or Light; I for Italic; C|E for Condensed or Expanded. Device independent fonts (outlines) omit {point-size}. The extension indicates the format (see below) of the font representation. For device-dependent (e.g., bit map) representations, the standard conventions contain no clue to the orientation, resolution, etc., of the font.

An alternate font naming convention is used to expand the code describing the type face, in Press files and in some of the formats below. This convention uses a three letter code, selecting from the orthogonal attributes (bold/medium/regular, italic/regular, condensed/regular/ expanded). In the TimesRoman10B example, this face code is thus BRR. See [Sproull4] for the precise mapping between this code and the actual numeric specifications used in Press and in font dictionaries.

Formats

The format of a file is indicated by its extension. Standard formats include:

.SF	Outline representations edited with FRED (device-independent).
.SD	Compact outline representations, produced by PrePress from .SF files (device-independent).
.AC	Raster representations edited or created with <i>PrePress</i> from SD format (device-dependent).

The remaining formats can be converted to AC format, and can be produced from AC format:

.CU	"Carnegie-Mellon" format raster representations.
.AL	Raster representations in Alto (CONVERT) format.
.STRIKE	Raster representations in Alto (BITBLT) format.
.EP	EARS-format portrait font (run-coded raster).
.EL	EARS-format landscape font (run-coded raster).

Example: Helvetical2LEP is a 12-point Helvetica font for EARS (face code MIR).

Resolution

While raster representations can be automatically converted from outline descriptions, this process is not performed efficiently with present hardware. In addition, some hand tuning of a scan converted font often improves its appearance. Therefore, each *Press* printer is expected to have local raster copies of all necessary fonts, converted to whatever resolution is appropriate. Although Press format allows users to include personal fonts in outline form, Spruce software does not implement this feature. The *Orbit* printers (*Dover*, *Sequoia*, *Pimlico* -- see [Ornstein] and discussion below) generally run at 384 dots per inch. *Dovers* currently running 350 dots will soon be converted to 384 (see issues and plans, below). The *Orbit* hardware supports a maximum of 4096 dots per scan line, and so limits printing length to 10.6" at 384 dots.

Widths

There is one problem with keeping fonts external to Press files: the width of a character is not specified locally, and formatting programs need to know character widths for things like text justification. This information is kept in a file named Fonts.Widths. Any changes to the width information (for purposes of standardization or aesthetic appeal) will alter the appearance of Press files which were produced using different widths. This is currently a major problem, as we attempt to reach compatibility with the printing world. Additional incompatibilities are the printing world's two kinds of spaces (numeric and regular), different characters for hyphen and minus, and the use of ligatures (multiple character combinations like "ae" or "ff") and kerning (sometimes called "optical spacing" -- altering inter-character spacing for specific pairs of characters).

A list of the standard fonts to be available on printers in PARC beginning June 15 is contained in an appendix. More may be added, to all or some, as time goes on.

Printing Programs

The *Sears* program [Rider1] operates the *Ears* printers as a server, spooling and printing EARS format files (using the EFIP protocol).

There are two programs that accept Press format files: *Press* [Sproull1] and *Spruce* [Swinehart1].

Press

This program, the older of the two, will accept and render nearly all Press specifications, including color (results depend on the printer). *Press* is a stand-alone program, whose intended application is the production of high-quality, high-resolution graphical images. *Press* is available on most of the printers described in the following section, except for the very fast ones (see *Printers*, below). *Press* does not operate as a "spooling" network service program, and is therefore not the printing program of choice for high-volume text printing.

Spruce

Spruce is intended as a printing service system, primarily for printing office documents. It accepts a subset of Press files including text characters, rectangles, and most Alto screen resolution bit maps. It does not handle splines or arbitrary bit patterns, nor does it honor brightness and color requests (but see the note below). Enterprising users could obtain limited line-drawing capability by creating and using appropriate "spline fonts". *Spruce* operates only on printers controlled by *Alto II/Orbit* hardware (see *Printers*, below).

AIS

Continuous tone images which have been digitized are stored as an *Array of Intensity Samples*. *AIS* format is used to store such images. Included in an *AIS* file are various data about the image which are important for accurate reproduction. Software is available which prints *AIS* files using a variety of halftoning techniques. Halftoning simulates various gray tones using only one ink color by varying the amount of ink in an area proportional to the original intensity. By halftoning several different color separations, a wide range of hues can be achieved.

Color Note

Press format allows specification of hue (color) as well as intensity. The *Press* program implements this feature directly on the *Pimlico* printers (see below). An interim measure has been used, in *Spruce* for *Pimlico* and in *Press* for the *Colorado* system (see below), to produce color images. In these systems, three or four Press pages, one per color (sometimes including black) are required for each printed page. A full conversion to direct color specification support is expected for *Spruce*.

Formatting Programs

Maxc Programs

The *Pspool* program [Fiala1] runs as a background job on the *Maxc* systems, formatting and transmitting documents to printers as they appear on the <PRINTER> directory. Files are placed there either explicitly or in response to the attempts of programs to write to the LPT: device. Files already in EARS or Press format may be sent to an appropriate printing server. Press format files may, alternatively, be converted to EARS format and sent to the *Ears* server. Pspool format may be converted into either EARS or Press format and sent (see the information on *Docgen*, below). *Pspool* will add necessary .EP or .EL files (EARS font descriptions) to a file on its way to *Ears*, if that information is not already in the file. It will use the *Doculist* section of the EARS file to decide what to send.

The *EPress* program is invoked by *Pspool* to translate Press format files into EARS format for printing on *Ears*. This conversion is automatic. The user has the option to save the resulting EARS file.

The standard way to direct text or pre-formatted files to printing servers is to run the *Ears*, *Press*, *List*, or *Copy* commands. These programs do intelligent things about selecting files with the proper extensions, embedding appropriate header information in EARS files, etc. Headings, page numbers, margin transformations and size reduction are also available through the *Ears* command.

The *Printer Status* command will indicate the names of files spooled on the <PRINTER> directory and of recently printed files, and the state of the most recently active printing server. *Respool* may be used to retransmit a file that was sent for printing but did not get printed correctly for some reason.

The *Pub* [Tesler] formatting program is a batch processing system that is useful in formatting large documents, especially if they need indices or cross-reference, or if they are going to be updated frequently. *Pub* will produce plain text output (for on-line perusal) or EARS format output, including a *Doculist* that *Pspool* can use to include the necessary fonts. Once the *Ears* system is gone, production of *Pub*-formatted documents will of necessity be a two-stage process, producing first the EARS format file on *Maxc*, then a Press format file on an Alto using *PressEdit* (below).

Docgen.Prt

Pspool often needs additional information about a user's printing preferences before it can perform its tasks. When sending Press files, for instance, *Pspool* must know what server to send them to. Although it is always possible to specify these options directly, it is useful to be able to default them. The system provides global defaults for each option. The user may specify different ones by creating a *Maxc* file called *Docgen.Prt*. The system defaults and the format of *Docgen.Prt* may be found in [Fiala2].

Alto Programs

The *Gears* program [Rider1] will produce an EARS format file from a text file, transmitting the result to *Ears*. It will optionally supply a heading and page numbers on each page, expand tabs properly, and obey a small number of font change and formatting specifications. In addition, it uses the *Ears* protocol to report on the status of the printer.

NGPR is part of the design automation package. It produces EARS format versions of Sil format files, either leaving them on the local disk or sending them to *Ears*.

The *Empress* [Boggs/Maleson/Tiberi] and *NPPR* programs are analogous to *Gears* and *NGPR*, except they produce Press files. *Empress* does not provide all the formatting options that *Gears* does, but it does include some useful Press file processing functions. It is in addition willing to ship files that are already in Press format to a Press printer. *NPPR* uses *Empress*, via a command file, to print its output.

The *OfficeTalk-Zero (OZ)* system [Newman/Mott] is also capable of producing and directly sending Press files to suitable printers.

The *Bravo* editor will produce hardcopy in either EARS or Press format. It can either produce a local file or transmit the result directly to a printer. In the latter case, *Bravo* uses the file *Swatee* as a scratch file. Recent versions of *Bravo* and *Draw* (see below) will accept color information and produce the appropriate "color separated" Press file.

User.Cm

The [HARDCOPY] section of the *User.Cm* file on one's Alto disk serves to personalize default printing options (e.g., preferred printing format, preferred printer, etc.) It is analogous to *Docgen.Prt* on *Maxc*. The following example contains a sample of each kind of entry that I am aware of:

[HARDCOPY]	
PREFERREDFORMAT: Press	Press or Ears
EARS: Palo	Name or legal network address of Alto providing Ears printing service
PRESS: Menlo	Similarly for preferred Press printer
COLOR-PRESS: Victoria	Preferred color printer (Press only)
PRINTEDBY: "\$"	See explanation below
FONT: TIMESROMAN 10 MIR	See explanation below

The FONT entry, used up to this point only by *Empress*, specifies that TimesRoman10i (italic) should be used as a default font instead of Gacha8 (*Empress's* default choice). The second, point size argument, and the third, face specification argument are optional. The face argument contains three letters specifying weight (M, B, or L), slope (R or I), and expansion (C, R, or E), respectively.

The PRINTEDBY field, if present, specifies the name to be used in the Name field on the break page. The current disk login name will replace the character \$. *Empress* and *Bravo* both chose "\$" as a default in the absence of a specification.

As far as we know only *Bravo* honors the PREFERREDFORMAT entry.

Of course, the [BRAVO] section of *User.Cm* also contains font selection specifications.

Related Programs

Markup [Newman1], *Draw* [Baudelaire1], and *Sil* [Thacker/Sproull] are interactive programs that produce Press file output for hardcopy. *Markup* and *Draw* are illustration programs with different strengths and emphases. *Sil* is a part of the hardware design automation package, but is also useful for producing charts and other simple line drawings.

PressEdit [Newman2] is used to edit and merge Press files (at the page level), and to convert EARS format files to Press format.

Fred [Baudelaire2] is an interactive spline curve editor specialized for font design. *PrePress* [Sproull2] is a utility program for converting spline character shapes to bit maps in various formats and resolutions, as described in the *Fonts* section. Both are of use mainly to implementors of printing or display software.

Printer Types

All the printers described below are *Raster Output Scanner (ROS)* printers. The complete printing facility includes a *ROS* and an *Electronic Image Processor (EIP)*, consisting of an Alto and either a little or a lot of specialized imaging and interface hardware. Newer systems interface *EIP* to *ROS* using the "9-Wire standard" [Rider2, Rider3] -- a standard hardware interface and communications protocol.

Ears. [Rider1] A modified 7000 duplicator engine with *Slot* (Scanned Laser Output Terminal) imaging optics. *Ears* is controlled by a very powerful special purpose *EIP*, the *Research Character Generator (RCG)*. The *RCG* is in turn controlled and supplied with data by an Alto. The *Ears* printer was designed and built at PARC.

Slot/3100. modified 3100 copier with *Slot* imaging optics. Image processing is done entirely in Alto microcode, and the machine is controlled through a very simple interface. This simple structure is made possible by the relatively slow operating speed of the 3100 engine. Because it runs slowly enough, and because the 3100 engine provides excellent solid-area development, high-quality, high-resolution graphic objects may be rendered on this machine. The *Slot/3100* systems were built by XEOS.

Dover. [Ellenby, Sproull3] A modified 7000 duplicator with an improved *Slot* head totally contained within the original engine. *Dovers* are engineered to be produced in reasonable quantities. They contain *ROS Adapter* (video and control electronics) modules that may be used in several other printers (see below). The adapters communicate with their controlling Alto IIs using the *9-Wire Standard*. A *Dover* Alto contains relatively inexpensive *EIP* hardware, called *Orbit* [Ornstein, Sproull3]. This device can, with some preprocessing assistance from its Alto, generate text pages and ruled forms at full *Dover* speed.

Dover's capabilities are largely determined by the bandwidth of the associated *EIP* and storage medium. The *Orbit* hardware reduces the bandwidth required to do text and solid rectangles, but *Alto* main memory and T80 disk bandwidth limitations restrict *Alto/Orbit/Dover* systems to relatively low-resolution graphical output. The *Alto/Orbit* hardware can produce high-resolution, high-quality graphical output when used with slower engines (e.g., *Sequoia*, *Pimlico*).

Dover and *Orbit* were developed by team from PARC and EOD/SPG.

Sequoia. A modified 3100 copier equipped with a *Slot* head. *Sequoia* uses the *Dover ROS* adapter, obeys the 9-wire standard, and is controlled by an Alto II-*Orbit*. *Sequoia* was developed and manufactured at XEOS. Its operating speed and good solid-area development allow *Sequoia* to render high-quality graphics.

Pimlico [Swager, Baudelaire/Swinehart] A modified 6500 color copier. The original optics have not been removed; therefore hybrid composition techniques involving both platen and raster-scanned inputs are possible. *Pimlico* uses the *Dover ROS* adapter, obeys the 9-wire standard, and is controlled by an Alto II-*Orbit*. The entire electronics control package has been replaced by a microprocessor-based system, using an M6800, that can communicate with the controlling Alto II via standard adapter commands. This vastly improves the engine control protocols and capabilities. *Pimlico* was developed by a team from PARC and EOD/SPG. Its development was expedited with special funding provided by C. Peter McColough as part of the Xerox World Conference effort.

Alto/Orbit/Pimlico is capable of high quality renditions of graphical objects.

Colorado. A modified 6500 color copier that has been in service at XEOS in Pasadena for several years. *Colorado* has been developed by Dale Green (OSL) and his group, and has been used in a variety of pioneering color copying and printing experiments. This system is a high

resolution, very high quality RIS-ROS system, with color correction, tone reproduction (TRC), and halftoning implemented in hardware. The ROS may also be used, at lower resolutions, as a color printer for the Alto. The interface to the Alto is the same as is used for the *Slot/3100*. *Colorado* uses *Press* and *AIS* printing software to print "color separated" files (see the note on color rendition above). Its 6500 engine has been augmented with a fourth developer (black) to print four-color pictures.

TC-200. A modified TC-200 telecopier, interfaced to an Alto in a manner similar to the *Slot/3100*. This system also contains RIS (raster input scanning) capabilities. It was developed at ADL. A TC-200 system will soon be available at PARC.

Versatec Plotter. A modified Versatec electrostatic plotter, interfaced to an Alto in a manner similar to the *Slot/3100*. This system is capable of printing on extremely large pieces of paper, but at a slower speed than the xerographic printers.

Performance and capability specifications for these printers appear in an appendix.

Impending Changes

Orbit Printer Resolution

We are currently operating all the *Orbit* printers (*Dovers*, *Pimlicos*, *Sequoias*) in Palo Alto at 350 dots/inch. For a variety of technical reasons, we plan to switch to a 384 dots/inch resolution as part of the June 15 cataclysm. With one important exception, users will not notice the change.

The exception is that, again for technical reasons, *Orbit* printers at 384 dots/inch are limited to a 10.6" high page. *Dovers* and *Sequoias* will thus enforce a .4" top margin, which may cause some information to be "clipped" from the page. *Pimlico* output will be unaffected.

The Demise of Ears

The *Palo/ Ears* system, because of its advancing age, is becoming quite difficult to maintain. For this reason, a second floor tribunal has decided that *Palo* will be removed from service on June 15, 1978. The remainder of this section discusses various issues that this decision raises.

Spruce printers are now functionally competitive with the *Ears* system, although the number of available, useful fonts is somewhat diminished, particularly landscape fonts (see the appendix). Pending enabling technology, we are delaying plans to fetch additional font specifications from less restrictive storage on Juniper or Ivy systems. Some performance improvements are also planned in anticipation of the increased load.

It was also necessary to decide the fate of EARS format files: those currently in existence, and those that will continue to be produced by *Pub*. Most EARS files, including all new *Pub* products, contain the directory entries (*Doculists*) that allow the *PressEdit* program to produce equivalent *Press* files from them. We hope it will be possible to convert EARS files that do not contain this directory information to those that do. If not, then a relatively small number of old EARS files, produced by older versions of *Pub*, will no longer be printable.

One additional impending change, the modification of some of our character widths, add to the difficulties of retiring *Ears*. This is further discussed in the following section.

Character Widths

Press format files do not in general specify the height and widths of their text characters. Instead, it is assumed that the producer of a Press file (e.g., a formatting program), and its consumer (usually a printing server) have a common notion of these values. Each *Alto* based formatting program uses the file Fonts.Widths, which describes the heights and widths of all available type fonts in device-independent units. This width file is carefully maintained by each installation. It is augmented as new fonts are added.

The printing programs maintain corresponding width information, usually device-dependent, that agrees with the information in Fonts.Widths. In this way, the accuracy of Press renditions is guaranteed.

Unfortunately, we have learned that the widths originally assigned to the characters in the font families TimesRoman and Helvetica were incorrect -- for the most part, the current widths produce excessive inter-character spacing. For this reason, it will be necessary to institute a new width standard, using the following scheme:

On June 15, 1978, a new Fonts.Widths file will be promulgated. At the same time, an improved set of fonts, geared to the new widths, will be installed on all Palo Alto *Spruce* and *Press* systems (many sites already use these new fonts and widths). Users who obtain the new widths file should notice no changes. Problems might, however, be expected in printing previously

created Press files (using the old width specifications). The following steps will be taken to reduce the inconvenience:

Old Press Files: The Press file document directory includes some spare fields. We have chosen one two-word field to represent a date and time, in Alto Standard format [Taft2]. The assumption is that old Press files contain values in this field that can be rejected as valid dates (typically 0 or -1). Subsequent to June 15, 1978, any Press file whose date appears invalid, or whose date precedes June 15 will be printed using the old widths. Files with dates of June 15 or later will use the new widths. It is not clear how long these old widths will remain available. The scheme to be used is quite general, and can support additional future width modifications, but such changes are painful, and are expected to be quite rare. *Press file implementors:* consult the information in [Swinehart2] or in the newly-updated version of [Newman/Sproull] for details required to update your output modules.

Old EARS Files: As indicated above, it should be possible to add Doculist directories to directory-less files. (These files contain the EARS fonts directly; PressEdit does not find them useful.) The old Fonts.Widths will be available, so that PressEdit may be used to create "old" Press files, complete with invalid or pre-June dates.

New EARS Files (Pub output): The .EP format files that Pub uses instead of Fonts.Widths to perform its formatting will be modified to reflect the new widths (note that only the width information in each font file will remain relevant, since Palo will have been retired.) Thus, PressEdit will have the proper information, using the new Fonts.Widths, to produce "new" Press files. These mechanisms are clumsy enough that we expect them to hasten the transition to newer formatting systems.

Longer-Term Issues

Many other modifications and additions have been contemplated for improving printing service. Most will require extensions to either the printing protocols or to the Press format.

Potential printing protocol extensions would permit the specification of the sending user (distinct from the creator of the Press file). It would also permit the transmission of non-resident, machine-dependent fonts (Press format supports the transmission of character shapes as outlines, but not as bit maps). In addition, one could request priority treatment, or could request that printing be delayed until the user is present, for security reasons. Other extensions might allow references to fonts and Press files on other file systems (e.g., Ivy) to be spooled, instead of the files themselves.

These issues are mentioned as possible areas of continued development; implementation should be considered distant.

Summary

This section has described some impending changes in terms of some problems and their solutions. The solutions are summarized in the *Key Points* section on page 1.

References

- [Baudelaire1] *Draw*, by Patrick Baudelaire, found in the *Alto User's Handbook*, updates in [Maxc]<Altodocs>Draw-News.Ears.
- [Baudelaire2] *Fred*, by Patrick Baudelaire, [Maxc]<Gr-Docs>Fred.Ears, 22 pp.
- [Baudelaire/Israel/Sproull] *Array of Intensity Samples -- AIS*, by Patrick Baudelaire, Jay Israel, and Robert Sproull, [Maxc]<AIS>AIS-Manual.Ears, 48 pp.
- [Baudelaire/Swinehart] *Pimlico Protocol and Software*, by Patrick Baudelaire and Dan Swinehart, [Maxc]<Dover>Pimlico.Press, or [IFS]<Pimlico>Pimlico.Press, 21 pp.
- [Boggs/Maleson/Tiberi] *Empress*, by David Boggs, Joe Maleson, and Rick Tiberi (D. Swinehart, ed.), [Maxc]<Altodocs>Empress.Tty, 5 pp.
- [Boggs/Taft] *FTP Reference Manual*, by David Boggs and Edward Taft, found in the *Alto User's Handbook*, or as [Maxc]<Altodocs>Ftp.Tty, 17 pp.
- [Ellenby] *Dover Overview*, by John Ellenby, available from the author.
- [Fiala1] *Document Distribution*, by Edward Fiala, [Maxc1]<Doc>Docudis.Ears (probably archived), 24 pp.
- [Fiala2] *Document Distribution*, by Edward Fiala, [Maxc1]<Doc>Docdis.Ears (probably archived), 8 pp.
- [Lampson] *Bravo, Reference Manual*, by Butler Lampson, found in the *Alto User's Handbook*.
- [Maleson1] *Pictures, PRESS, and you*, by Joe Maleson, archived on [maxc]<Press>HalftoneMemo.Press and HalftonePlates.Press.
- [Maleson2] *Rotating halftone screens*, by Joe Maleson, archived on [Maxc]<Press>Rotation.Press.
- [Newman1] *Markup Reference Manual*, by William Newman, found in the *Alto User's Handbook*, or as [Maxc]<Altodocs>Markup.Ears, 11 pp.
- [Newman2] *PressEdit Reference Manual*, by William Newman, [Maxc]<Altodocs>PressEdit.Tty, 1 pg.
- [Newman/Mott] *OfficeTalk Zero Reference Manual*, by William Newman and Tim Mott, [Maxc]<SSLDocs>Oz-15.Ears, 25 pp.
- [Newman/Sproull] *Press File Format*, by William Newman and Robert F. Sproull, [Maxc]<Gr-Docs>PressFormat.Press, 16 pp.
- [Ornstein] *Orbit General Description*, by Severo Ornstein, [Maxc]<Dover>OrbitWriteup.Press, 38 pp.
- [Rider1] *Ears, Gears, Sears, and Other Related Items*, by R. E. Rider, [Maxc]<Altodocs>Gears.Tty, 13 pp.
- [Rider2] *ROS Interface Conventions*, by R. E. Rider, [Maxc]<Rider>RosInterface.Press, 4 pp.

- [Rider3] *RIS/ ROS Interface Conventions*, by R. E. Rider,
[Maxc]<Rider>RisRosInterface.Press, 5 pp.
- [Shoch1] *A File Transfer Protocol Using the BSP -- 2nd Edition*, by John Shoch, [Maxc]<Pup>FtpSpec.Ears, 16 pp.
- [Shoch2] *EFTP: A PUP-based Ether File Transfer Protocol*, by John Shoch,
[Maxc]<Pup>EFTPSpec.Ears, 16 pp.
- [Sproull1] *Press Printer Operation*, by Robert F. Sproull,
[Maxc]<Gr-Docs>PressOps.Ears.
- [Sproull2] *PrePress Manual*, by Robert F. Sproull,
[Maxc]<GR-DOCS>PrePress.Ears, 17 pp.
- [Sproull3] *Programmer's Guide to Orbit, the ROS Adapter, and the Dover Printer*, by Robert F. Sproull, [Ivy]<Spruce>OrbitGuide.Press, 37 pp.
- [Sproull4] *Font Representations and Formats*, by Robert F. Sproull,
[Maxc]<GR-DOCS>FontFormats.Ears, 21 pp.
- [Swagger] *Gandalf (Pimlico) Overview*, by Gary Swagger, not yet released.
- [Swinehart1] *Spruce Reference Manual*, by Dan Swinehart,
[Ivy]<Spruce>SpruceManual.Press, 13 pp.
- [Swinehart2] *Let's Talk About Printing*, by Dan Swinehart, [Maxc]<Gr-Docs>PrintingUpdate.Press, 2 pp.
- [Swinehart3] *Printing at Palo Alto*, by Dan Swinehart (this document), [Maxc]<Gr-Docs>Printing.Press, 23 pp.
- [Taft1] *Ears Protocols*, by Edward Taft, [Maxc]<Pup>EarsProtocols.Bravo, 1 pg.
- [Taft2] *Alto Time Standard*, by Edward Taft, [Maxc]<Taft>AltoTime.Bravo, 4 pp.
- [Tesler] *Pub: The Document Compiler*, by Larry Tesler, Stanford Artificial Intelligence Laboratory Operating Note 70, September, 1972.
- [Thacker/Sproull] *Sil, Analyze, Gobble, Build Reference Manual*, by C.P. Thacker and R.F. Sproull, [Maxc]<SIL>SILManual.Press, 27 pp. + appendices

Appendix -- Font Naming Conventions, User.Cm Formats

Font file naming convention:

{family-name-in-full}{point-size}{[B|L]}{[I]}{[C|E]}.{extension}

B|L: Bold, Light
 C|E: Condensed, Expanded
 Point Size: Omitted for device-independent fonts (outlines)
 Extension: Designates encoding format (see below).

Examples: TimesRomanB.SD, TimesRoman10B.AC

Type face encoding convention (in Press files, font specifications):

B M R:	Bold, Medium, Light	2, 0, 4 (see [Sproull4])
I R:	Italic, Regular	1, 0
C E R:	Condensed, Expanded, Regular	6, 12, 0

Font formats, identified by file extension:

.SF	Spline outlines, represented by text descriptions.	
.SD	Compact outlines, produced by PrePress from .SF files (device-independent).	
.AC	Raster representations edited or created with PrePress from SD format (device-dependent).	
.CU	"Carnegie-Mellon" format raster representations.	(device-dependent,
.AL	Raster representations in Alto (CONVERT) format.	convertible to/from
.STRIKE	Raster representations in Alto (BITBLT) format.	.AC format)
.EP	EARS-format portrait font (run-coded raster).	
.EL	EARS-format landscape font (run-coded raster).	

Example: Helvetica12LEP is a 12-point Helvetica font for EARS (face code MIR).

User.Cm, [HARDCOPY] Section (Comprehensive example)

[HARDCOPY]	
PREFERREDFORMAT: Press	Press or Ears
EARS: Palo	Name or legal network address of Alto providing Ears printing service
PRESS: Menlo	Similarly for preferred Press printer
COLOR-PRESS: Victoria	Preferred color printer (Press only)
PRINTEDBY: "\$"	See explanation below
FONT: TIMESROMAN 10 MIR	See explanation below
FONT:	Overrides Empress's default (Gachaß). Uses Face codes as given above.
PRINTEDBY:	Overrides Empress, Bravo defaults (Username on Alto disk). The "\$" character requests the Username field.
Others:	See text (see also the [BRAVO] section of <u>User.Cm</u> .)

Appendix -- Printer Characteristics and Capabilities, Current Palo Alto Printers

Printer Specifications

Printer	Paper in/sec	Speed sec/page	Orientation	Resolution dots/inch	Software	Notes
Ears	10	1	Landscape ¹	500x500	Sears	Predominantly text
Slot/3100	3.4	4	either	384	Press ⁶	High quality graphics
Dover	10	1	Landscape	300-400	Spruce	Predominantly text, low res. bit maps
Sequoia	3.4	4	Landscape	300-400	Spruce ³	Low vol. text, quality graphics
Pimlico	4	18 ⁷	Portrait	300-400	Press ²	High quality color graphics
Colorado	4	18 ⁷	Portrait	300-700 ⁴	Press	High quality 4-color gr., cop.
TC-200	.5 ⁵	22 ⁸	Portrait	200	Press	Low cost, RIS capability
Versatec	--	--	Portrait	--	Press	

Notes:

1. Paper moves sideways through machine. Paper moves longways in Portrait mode.
2. Spruce is also available for this system.
3. Press may be available soon for this system, as well.
4. High resolution used in direct copying (RIS/ROS) operation, and in special printing modes, using two printer dots per dot specified by the image processor.
5. Variable speed, depending on black/white ratio, line by line.
6. The AIS program will also run on all Press printers.
7. About 6 seconds per revolution, three revolutions per full-color page. The first page takes another several seconds to emerge from the machine.
8. Roll fed machine -- no inter-page spacing.

Summary of Composition/Printing Program Capabilities

Editor/Illustrator Outputs					Press Features	Printing Software & Printers ¹			
Bravo	Markup	Sil	Draw	AIS		Spruce	Press (D)	Press (S,P)	Ears
x	x	x	x		text	x	x	x	x
x		x	x		rectangles	x	x	x	x
			x		graphics font		x	x	
			x		objects			x	
	x				low res. dots	x	x	x	x
				x	high res. dots			x	
x ²		x	x	x	color	p ²		P	

¹ D=Dover, S=Sequoia, P=Pimlico; ² "Color-separated" Press files, only, at present

Currently Available Printers

Name	Ether Address	Location	Printer Type	Configuration	Server Protocol
Palo	3 #3 #	35-2077	Ears	Model 44	EFTP (<u>Ears format</u>)
Menlo	3 #164 #	35-2026	Dover	2 Model 31's	EFTP (<u>Press</u>)
Wonder	6 #265 #	35-3040	Dover	Model 31 only	EFTP (<u>Press</u>)
Clover	3 #121 #	35-2069	Dover	2 Model 31's	EFTP (<u>Press</u>) or stand-alone
Viola, Victoria	3 #341 #	35-2069	Pimlico	T80	EFTP (<u>Press</u>) or stand-alone
Kanji	5 #52 #	34-B10	Dover	T80	EFTP (<u>Press</u>)
Daisy	5 #6 #	33-249	Dover	2 Model 31's	EFTP (<u>Press</u>)
Slot-2	3 #116 #	35-2015	Slot/3100	T80	stand-alone, by appt. only

Appendix -- Available Fonts (Palo Alto)

Files Containing Available Spruce Fonts (as of June 15, 1978 -- PARC only)

[Ivy] <Dover>Spruce.Fonts

Available Spruce Fonts (as of June 15, 1978 -- PARC only)

<i>Family</i>	<i>Size (pt.)</i>	<i>Face</i>
TimesRoman	8, 10, 12	MRR, BRR, MIR, BIR
Helvetica	7, 8, 10, 12 6 18	MRR, BRR, MIR, BIR MRR, BRR MRR, BRR
Gacha	8, 10 12 12	MRR, MIR MRR MIR
Cream	10, 12	MRR, BRR, MIR, BIR
Math	8 10	MRR MRR
Apl	8, 10	MRR
Hippo	8 10	MRR MRR
Sigma	20	MRR
Sail	8	MRR
Gates	32	MRR
Logo	24	MRR
Keyhole	20	MRR
Dots	256mica	MRR (for Alto bit maps)
Arrows	10	MRR

Landscape Fonts:

TimesRoman	8	MRR, MIR, BRR
TimesRoman	10	MRR, BRR
Helvetica	6	MRR, MIR
Gacha	8	MRR
Sail	6	

Appendix -- Useful File Directories

<i>System</i>	<i>Directory</i>	<i>Description</i>	
[Maxc1]	<Gr-Docs>	Graphics-related documents: <u>Press</u> formats, font formats, PrePress and Fred documentation, etc.	
	<Fonts>	.EP, .EL, and (obsolete) .AL fonts for use on Altos and <i>Ears</i> . As of this writing, they correspond to the "old" <u>Fonts.Widths</u> specifications. Alto users should obtain their .EP and .EL fonts (for use with Bravo) from this directory.	
	<AltoFonts>	.AL, .STRIKE fonts for use on Altos. As of this writing, they correspond to the "old" <u>Fonts.Widths</u> specifications. Alto users should obtain their screen fonts from this directory.	
	<PressFonts>	.SF (mostly archived), .SD, and .AC fonts that are sources for the <Fonts> directory, and for current Palo Alto <i>Spruce</i> and <i>Press</i> fonts.	
	<Alto>, <AltoDocs>	The repository for many of the systems described in this document. The current "old" <u>Fonts.Widths</u> also resides on <Alto>.	
	<Dover>	Documents and programs for the <i>Spruce</i> system, the <i>Orbit</i> EIP, and the <i>Dover</i> , including diagnostic programs. Many of these files are obsolete, having been superseded by files on [Ivy]<Spruce>.	
	<AIS>	The AIS program, manual, and some sample AIS pictures.	
	<Press>	The Press program operations manual, many sample Press files.	
	<Ears>, <Graphics>	Nearly empty, mostly obsolete directories.	
	[Ivy]	<Spruce>	Current <i>Spruce</i> system and operations manuals, the current <i>OrbitTest</i> diagnostic program, other systems valuable for <i>Spruce</i> development.
		<PressFonts>	.SD, .AC, and .AL fonts corresponding to the proposed "new" <u>Fonts.Widths</u> standards. These have also been scan-converted at 384 dots/inch for <i>Spruce</i> and <i>Press</i> systems.
<Dover>		<i>Spruce</i> font directories, for use on <i>Dover</i> systems. Contains both "old" style fonts and some of those to be used as "new" style fonts.	
<Pimlico>		"new" style <i>Spruce</i> font directories for <i>Pimlico</i> printers.	
<Sequoia>		"new" style <i>Spruce</i> font directories for <i>Sequoia</i> printers.	
<Press>		Copy of [Maxc]<Press> as of 1 November, 1977.	

Appendix -- Glossary of Terms

AIS. *Array of Intensity Samples.* A file format, picture processing software, and printing program for arbitrary digitized (scanned) images. The *AIS* printing program has about the same bandwidth requirements as the *Press* program, and is in general supported on those printers that support *Press*.

bit map. a sequence of bits (usually represented as a sequence of computer words) that describe a dot pattern, at some specified resolution, simulating some graphic entity. To interpret a bit map one needs to know the order in which the sequence of bits must be laid down to produce the picture. This is usually expressed in terms of line length, the bit direction within each line, and the direction in which lines are scanned.

Bravo. An Alto-based text editor and formatter. *Bravo* can produce both Ears and Press format versions of its files (including "color separations" for *Pimlico*). It can either transmit these directly or retain them as named files.

BSP. *Byte Stream Protocol*, a high-level *PUP*-based protocol for high-speed, reliable transmission of information, in byte units.

Clover. An *Alto/Dover/Spruce* system, currently located in CSL.

Colorado. An experimental color RIS/ROS copier/printer system based on the 6500 color copier. Colorado comprises sophisticated digital and analog signal processing equipment, as well as an Alto interface (resembling that on the SLOTT/3100). It is located in Dale Green's laboratory at XEOS in Pasadena.

Doculist. A directory added at the beginning of an EARS format file specifying the fonts needed to print that file. The *Pspool* system removes this and includes the fonts themselves as a file is transmitted to the *Ears* printer. *PressEdit* uses this information to convert EARS files to Press format. The current *Pub* system produces files containing Doculists instead of actual fonts.

Dover. A ROS based on the 7000 duplicator and SLOTT optics, using the standard TTL ROS adapter. Its EIP, interfaced using the 9-wire standard, is an Alto II/Orbit (see text). *Dover* is intended for field experiments requiring relatively low-cost printing facilities.

Draw. An interactive Alto program using cubic splines to express straight and curved lines, in monochrome or color. *Draw* produces Press output for printing.

EARS: A file format containing text to be printed, formatting specifications, and font information.

-- A powerful, prototype EIP/ROS system, controlled by an *Alto* (or a *Nova*), specialized for printing text. The *Ears* system is a server system that accepts and prints EARS format files, without any preprocessing. (see *Sears, Palo*).

Eftp: A simple, *PUP*-oriented protocol, designed for file transmission from user programs to servers (especially printing servers). The server must acknowledge each packet before the next is sent. This protocol admits to compact implementation in user programs, offset by some reduction in bandwidth.

-- Programs written for *Alto* and *Maxc*, implementing both EFTP sending (user) and receiving (server) protocols. *Empress* (*Alto*) or *Press* (*Maxc*) are usually preferred for communicating with *Spruce* servers.

EIP: *Electronic Image Processor.* A device that converts image descriptions into raster information for printing on a *ROS*. Similarly, a device that processes *RIS* input rasters, storing and/or interpreting them. Example: *Alto II/Orbit*.

Empress. An Alto program which converts ordinary text files to Press format and sends them to a *Press* printing server. Empress will also transmit pre-formatted Press files without additional conversion.

Epress. A Maxc program, invoked by *Pspool*, that converts Press format files to EARS format for transmission by *Pspool* to an *Ears* system.

fonts. Character representations (either outlines or bitmaps).

Fred. A font design program on the Alto, which allows the specification of spline outlines for characters.

Ftp. A *File Transmission Protocol*, based on the *BSP PUP*-based protocol. *Ftp* is the preferred file transmission protocol, yielding quite acceptable bandwidths when correctly implemented. *Maxc* and *IFS* systems support this protocol; for *IFS*, it is the only file transmission currently supported.

-- An Alto program providing file transmission facilities using the *Ftp* protocol. This program also provides a "Telnet" facility, for standard Ascii terminal communication with a variety of systems on the Internetwork.

Gears. An Alto program which composes text files and transmits them to the *Ears* *ROS* system.

half-tones: Binary images which simulate continuous tone images by varying the number of dots printed proportional to the intensity.

Icarus. An interactive, display based LSI design and layout program. *Icarus* produces *Press* files as a means of obtaining "proof" hardcopy of the new circuits.

Internetwork. The collection of *Ethernets*, telephone lines, and gateway machines that comprise the current Xerox experimental network. All machines in the internetwork share a common, two-level address space. In general, any machine may communicate with any other, using one of the *PUP*-based protocols. Other packet protocols are not forwarded by gateways, limiting such communications to their local network of origin.

Kanji: An *Alto/Dover/Spruce* system, currently located in OSL (Bldg. 34).

Markup. An Alto program which allows text and graphic composition (at Alto screen resolution). Input and output is a Press file.

Menlo: An *Alto/Dover/Spruce* system, currently located in SSL.

Ngpr. An Alto program that converts Sil format files to EARS format, then (optionally) transmits the resulting file to an *Ears* server.

Nppr. An Alto program that converts Sil format files to Press format. The user must transmit the resulting Press file to a printing server.

objects (Press). Graphic data specified by an outline consisting of a series of display commands (*Moveto*, *Drawto*, *DrawCurve*).

Orbit. A hardware device connected to an Alto whose function is to place raster

representations of characters at specified positions on the bitmap for a printed page. An Alto/Orbit combination comprises an EIP.

OZ The Officetalk Zero system, which (among other things) produces file output in Press format.

Palo: An *Alto/Ears/Sears* system, located in CSL.

Pimlico. An obsolete name for a computer controlled ROS version of the Xerox 6500 color copier. (cf. Pimlico)

PrePress. A program for converting among the various font formats, and maintaining dictionaries of fonts.

Press: A file format allowing total specification of the printed appearance of a document. Press format can handle text, (half tone) bit maps, rectangles, and object outlines. It has provisions for inclusion of additional information, for use by text editors, formatting systems, etc.

-- An Alto program for printing Press format files on a variety of ROS printers (see the text and the appendices for a complete list).

-- A command provided by *Maxc* systems for printing text and Press files.

PressEdit. A program which runs on *Maxc* and converts EARS files to Press format, adds fonts to Press files, creates blank Press files with any set of fonts, extracts pages from EARS/Press files and puts them into a new Press file, merges figures with text documents, and will add the private data stamp to Press files.

Pspool. A server job on *Maxc* systems that formats and transmits (converted) text, EARS format, and Press format files to appropriate printing servers. Files are placed on the <Printer> directory by other programs; in addition to printing the files that appear there, *Pspool* manages this directory.

-- An augmentation to standard Ascii text format, specifying simple margin adjustments, font changes, and heading requests. The *Pspool* program interprets this format, converting to the proper printer specifications.

Pub. A program on *Maxc* which takes documents containing special formatting commands, and produces EARS files.

PUP. The *PARC Universal Packet*, a format for data packets that can be routed or broadcasted among any of the hosts in the internetwork. *PUP* format specifies only source, destination, error checking information, a top-level command code, and limited sequencing information. Additional protocols must be implemented by structuring the data contained in the packets.

PUP-based Protocols. Any of the hierarchy of communications protocols that have been defined in terms of *PUP* (e.g., *EFTP*, *FTP*, *BSP*, *RTP* (*Rondesvous/Termination Protocol*), etc.)

rectangles (Press). Special Press objects whose shapes are specified by two corners; the rectangle sides are drawn parallel with the paper edges. Rectangular shapes which are not parallel to the paper edges are considered regular objects.

ROS,RIS. Raster output(input) scanner. A device which produces a video stream, either to a printer (ROS) or from a document (RIS).

ROS Adapter. A hardware module which communicates among the *Orbit*, ROS, and printing engine.

Sears. The printing and spooling program for the *Ears* ROS system.

Sequoia. A Xerox 3100 copier with *Orbit* and ROS adapter.

server. A program which runs unattended, accepting input from the internetwork.

SIL. A design program on the Alto, which supports circuit design. The graphic features included for circuit design are useful for other design functions; SIL is sometimes used as a general graphic editor.

Slot. A laser ROS (*Scanned Laser Output Terminal*).

Slot/3100. A Xerox 3100 copier with a simple Slot interface.

Slot-2: An *Alto/Slot/3100* system, currently located in SSL (appt. only).

solid-area development. The ability to produce an image with large areas of solid toner (3100, 6500). Attempts to print solid areas on other copiers causes "white-out," where only the edges of the area are printed.

splines (as used in *Press*). A kind of curve definition, using cubic equations.

Spruce. Software for printing simple *Press* files (text, fixed resolution bitmaps) on an *Orbit* device. Spruce is a server.

Stand-alone printers. Printers which are not servers.

TC-200. A 200 dot per inch RIS/ROS system. Output speed is about 30 seconds per page.

Turkey. An *Alto/Sequoia/Spruce* system, located in SDD/Palo Alto.

Viola. An *Alto/Pimlico* system, located in CSL. Runs *Spruce* or *Press*.

Versatec. A matrix printer.

Wonder. An *Alto/Dover/Spruce* system, currently located in Bldg 35, 3d floor.

3100. A solid-area development Xerox copier.

6500. A solid-area development color Xerox copier.

7000. A high speed Xerox copier/duplicator.

↑

G Y P S Y of May 12, 1978

Quit
Scan for {} All Substitute {} for {} Length Time

⌘
A

G Y P S Y D E V E L O P M E N T

Find the Folder labelled...

System Files

Insert a new Folder labelled...

Execute {Put} Execute {Ftp Maxc} Execute {Ftp Ivy}

Execute {Chat} Execute {} Execute {}

HARDCOPY SUBSYSTEM

Hardcopy is a program for automatically invoking the Bravo "hardcopy" command from the Alto executive. It will also call FTP if necessary to retrieve files from a shared file server such as Maxc or IFS. This subsystem saves keystrokes and your concentration, but not clock time.

How to use it...

You can say to the Alto Executive:

```
> Hardcopy file-descriptor file-descriptor ...
```

where a file-descriptor has the usual format:

```
[host]<directory>fileName.extension
```

All fields are optional, except fileName. If ".extension" is omitted, a default is taken from the user.cm file. If "[host]" is omitted, Hardcopy looks first on the local Alto disk; if the file is not there, it takes a default remote host from the user.cm file. Once a "host" or "directory" appears on the command line, it remains in effect along the line until overridden by another. Version numbers and sub-directories can be used when contacting servers that support these features.

For multiple copies, attach a switch to a file-descriptor; for example "HardCopy someFile.bravo/3", for three copies. The copy quantity remains in effect along the command line until overridden by another.

A temporary file Hardcopy.scratch\$ is created and deleted if a file is retrieved from a server.

You need the proper fonts, and Bravo 6.0 or later.

Name completion and "*" substitution work for local files only.

How to obtain it...

1. Using FTP, load the file hardcopy.dm. This includes two files: hardcopy.run and UserCm.slice.
2. UserCm.slice contains two parts: a Bravo H.INIT macro and a specification of default host and default file extension. Edit the first part into the bravo portion of your user.cm file. Add the second at the end of user.cm, after replacing the host name and the extension with your favorite ones (the extension must include the leading dot). Don't forget to "Bravo/I".

For Xerox Internal Use Only -- May 22, 1978

SoftBitBLT

May 22, 1978

1

Soft BitBLT

This package contains a single procedure, BitBlt, which emulates the BitBlt instruction in software. It is not reentrant.

BitBlt(bbt)

bbt points to an even word aligned BBT structure as defined in BitBlt.decl. See the Alto hardware manual for details.

BitBlt does some setup in BCPL and then calls an assembly language procedure to do the work. It is distributed as three files:

BitBlt.decl	Declarations needed to use the package
BitBltB.br	BCPL setup code
BitBltA.br	Assembly language inner loop

Whole ALTO World Newsletter

Technology and Tools

XEROX

June 30, 1978

SPECIAL ANNOUNCEMENTS

WHOLE ALTO WORLD CHAIRMAN CHANGE - Liz Bond, present chairman of the Alto group, suggested at the June 1st meeting that a new chairman be selected for the Whole Alto World. Instrumental in forming the group and staffing the coordinator function, she has held the position since the group's inception. Liz will continue to participate actively as a representative from XEOS.

A committee, consisting of Art Axelrod (WRC), Liz Bond (XEOS), Terry Haney (SPG), Darwin Newton (GINN), and Dick Sonderegger (SDD), was appointed to nominate a successor.

RENEW YOUR SUBSCRIPTION NOW - As all periodicals must, it is time to validate the current Newsletter distribution list. Appended to the Newsletter is a form to be completed and returned. Please fill in your name, mailing address, and organization. There are also some optional questions concerning the Newsletter and how you use the Alto. The answers will be used to shape the information content of future editions. Return the completed form to the coordinator as indicated on the form. *Only those individuals that return a completed form will continue to receive the Newsletter.* For all others, the Newsletter is stored as a Press file and may be retrieved from <AltoDocs>WawNewsM-YY.press (this issue is WawNews6-78.press).

A special thanks to all of you who have taken the time in the past to pass along changes of address and/or terminations. It is greatly appreciated.

THE BIG FONT CHANGEOVER - The replacement of existing Times Roman and Helvetica fonts has begun. Pasadena and Palo Alto printers have completed the change, El Segundo should be changed by the time you read this, and WRC will soon do like-wise. Users should watch for local announcements.

Pre-existing documents will use the new character shapes with the old character widths, so they will still be right justified. New documents created by users who have not updated their disks will also use the new shapes and old widths with the same result. New documents created with up-to-date software using both the new character shapes and widths, will have the best appearance.

To update your disk retrieve and execute <Alto>PrintingUpdate.cm (but not before the printer update is announced at your location). Your disk will get a new Fonts.widths, EMPRESS, and PRESSEEDIT. You should also separately retrieve the new BRAVO 7.2 and DRAW.

GENERAL NOTES

A NEW FOLKLORE DOCUMENT - Roy Levin has written an introductory document, "A Field Guide to Alto-Land, or Exploring the Ethernet with Mouse and Keyboard". Intended for people that will do programming on the machine, it explains many of the often unspoken assumptions about the environment. Though much of it is PARC related (it's a PARC folklore document), it appears to be of general interest for anyone doing software development. It also contains an extensive glossary and a reference to key documents. A copy of the document can be retrieved from [MAXC]<Levin>FieldGuide.press.

Whole ALTO World Newsletter

2

WHOLE ALTO WORLD MEETING - The Whole Alto World meeting was hosted by Dick Sonderegger and SDD at the Cockatoo Inn in Hawthorne on June 1, 1978. Forty people attended, representing both long established and potential Alto using groups.

Doug Stewart of SPG outlined current and projected build activity. The 7th Alto build and the Dover build are nearing completion with a 8th build to begin in early November. As a result of increased activity, Ron Cude has been appointed as head of the Test and Checkout activity. Ron will also handle items returned for repair.

Jim Hall's group is now offering to maintain Altos for \$132 per month. This price revision results from additional maintenance experience and changes to the maintenance policy, e.g. the users provide the spares inventory for their own machines. Interested groups should contact him for details and quotes on additional Alto-related items.

Alto gateways should become operational by the end of the summer. The gateway can be configured using either with EIA boards or a communications processor. Using the EIA boards results in a lower initial cost but a higher incremental cost as new communications lines are added. Software for both is being developed jointly by PARC and SDD.

Bruce Malasky announced the release of MESA 4.0 and described some of the changes and enhancements. Complete MESA documentation is available on the SDD Ivy servers, Iris and IFS-2 under <MESA>Doc>.

Operation of the enhanced Gypsy was described by Frank Ludolph. The changes permit Gypsy to serve as both programming text editor and system executive, leaving more disk space for programming use.

Jerry Elkind, chairman of the Special Products Allocation Committee, discussed the Committee's operation. Composed of representatives from Corporate Research, ITG, and IPG, it allocates Xerox-built special products in limited supply, such as the Alto and peripherals, to requestors in a manner that best serves the Corporation.

Much of the afternoon session was used by organizational and site representatives to exchange information about current activities with others.

The meeting ended with a presentation by Ron Rider on printers under current development.

TOOLS

MAINTENANCE NOTES

BOARD REPAIR - Items returned to SPG for repair should now be sent to Ron Cude, M1-38, instead of Terry Haney as in the past. Items too large to be mailed should be sent to M1 North Dock, 555 S. Aviation, attn: Ron Cude. Please include all available information about the problem and symptoms, including hardcopy output if the item is a part of printing hardware. Repair will be scheduled in conjunction with SPG's other activities.

DISK DRIVE CHECKOUT BY THE USER - Users can now verify correct operation of their Alto's disk drive with DIEX, the DIablo disk EXerciser. To use it, ready a scratch disk (*any information on the disk will be destroyed*), etherboot, and enter diex^{CR}. DIEX will ask you to disable the writeprotect by entering '←'. In the upper window is a menu of commands and parameters. The parameters are preset to default values. If the disk you have inserted is a new disk,

i.e. it hasn't been previously initialized with a file system, bug *Init Verify* using the left (top) mouse button. When that operation completes, or if the disk has been previously used, bug *Do Test*. The test will now run for some time, alternately writing and reading the entire disk. Any errors detected will appear in the larger, lower window. If errors are indicated, contact your maintenance personnel.

Trident disk drives can be checked out in a similar manner using TRIEX.

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local IVY server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [XEOS].

MESA 4.0 - The newest MESA, version 4.0, was released on June 1st. The emphasis on this release is in three areas: implementation of features required for effective use of the new machine architecture, reduction of overhead in the basic system structures and improved performance of the Mesa runtime environment, and extension of the debugger's capabilities (primarily an interpreter for a subset of the Mesa language). To become a formal MESA user, with all attendant privileges, send a message to <Ludolph> describing your intended use.

USING GYPSY - Some questions have been posed about GYPSY files as a result of the enhancements announced in last month's Newsletter. To move BRAVO files to GYPSY, run DE-TRAILER <filename> then REFORM <filename> (both are available on [MAXC]<Alto>). DE-TRAILER removes all formatting information leaving a clear text file. REFORM, normally used to clean-up a malformed GYPSY file, reformats the file for GPYSY use. The resulting files will accept bold and italic formatting.

GYPSY can edit both formatted and unformatted documents. Those created by selecting "Fetch Working Draft" are, by default, formatted. To produce a clear text (unformatted) file, such as a command file, run the formatted document through DE-TRAILER. Subsequent editing will leave it unformatted.

NEW RELEASE: BROWNIE - This new program by Bruce Malasky is used to manage directories on file servers, especially the problem of distributing public files to various hosts. It is primarily for use by IFS administrators. The documentation is appended to the newsletter.

NEW RELEASE: DE-TRAILER - This subsystem strips formatting information from BRAVO and GYPSY files producing clear text files. Execute DE-TRAILER <filename>. There is no documentation.

NEW RELEASE: DIEX - This subsystem by Roger Bates is a Diablo disk diagnostic similar in nature to TRIEX used on the Trident. Instructions on its use are displayed on the screen at start-up. (Also see MAINTAINER'S NOTES above). DIEX can be booted off of the Gateway.

NEW RELEASE: ETHERRCVR - This package, by David Boggs, is for use by diagnostic programs that wish to exercise as many tasks as possible in attempting to provoke machine failure. It copies every packet it hears into an internal buffer. The documentations is attached.

NEW RELEASE: FIRST - Bob Dattola, WRC, is releasing FIRST, a document retrieval system, for experimental use on Altos. It accepts English language queries (sentences, phrases, or words),

ignoring the common non-content bearing words and reducing suffix variations (stemming) of the remaining words. The result is then used to search against a database of abstracts, computing a similarity function and ordering the resulting matches. It can be used to construct personal databases or to search an existing FIRST database of *Communications of the ACM* article abstracts, 1970 to 1977. For details, see the attached documentation. The software and database is available from both [XEOS]<Alto>First and [WRC] as indicated.

ReReleases - Subsystems

BRAVO - Version 7.2 fixes minor bugs and supports the new printing software. Retrieve and execute <Alto>BRAVO.cm.

DRAW - The latest release includes changes for the new printing software. Load <Alto>DRAW.dm.

EMPRESS - This version incorporates changes required by the printing software. Retrieve <Alto>EMPRESS.run. The documentation is unchanged.

NETEXEC - The 'Keys' and 'Host' commands have been combined into a single command 'FileStat'. The change is documented in the revised documentation <AltoDocs>NETEXEC.tty. It is not necessary to retrieve the boot file.

PREPRESS - The new version implements the new printing features and also includes a new user interface, invoked by calling PREPRESS without parameters. Retrieve <Alto>PREPRESS.run. The documentation is unchanged.

PRESSEEDIT - The latest release conforms to the new printing requirements and adds a new facility to add page numbers to a press-format document. Retrieve <Alto>PRESSEEDIT.run. Revised documentation is available on <AltoDocs>PRESSEEDIT.tty.

TECHNOLOGY

This month's paper by Terry Winograd, a member of the Sanford faculty and a consultant to Xerox, questions our current programming methodology and suggests an alternate approach. As he states in the abstract, "... we need to shift out attention away from the detailed specification of algorithms, towards the description of the properties of the packages and objects with which we build."

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC]<AltoDocs>WawNewsM-YY.press or may be obtained from the editor, Frank Ludolph, XEOS, by messaging <Ludolph> or calling Intelnet 8*923-4356.

Why Programming Languages Are Obsolete

Terry Winograd
Stanford University

Abstract

As computer technology matures, our growing ability to create large systems is leading to basic changes in the nature of programming. Current programming language concepts are not adequate for building and maintaining systems of the complexity called for by the tasks we attempt. Just as high-level languages enabled the programmer to escape from the intricacies of a machine's order code, higher-level programming systems can provide the means to understand and manipulate complex systems and components. In order to do this, we need to shift our attention away from the detailed specification of algorithms, towards the description of the properties of the packages and objects with which we build. This paper analyzes some of the shortcomings of programming languages as they now exist, and lays out some possible directions for future research.

Introduction

Computer programming today is in a state of crisis (or, more optimistically, a state of creative ferment). There is a growing recognition that the available programming languages are not adequate for building computer systems. Of course, as any first year student of computation theory knows, they are logically sufficient. But they do not deal adequately with the problems we face in our day-to-day work of programming. We get swamped by the complexity of large systems, lost in code written by others, and mystified by the behavior of our almost debugged programs. When we look to the integrated multi-processor systems that will soon dominate computing, the situation is even worse.

This crisis in software production is far greater (in overall magnitude at least) than the situation of the early 50's that led to the development of high-level languages to relieve the burden of "coding." The problems are harder to solve, and the costs of not solving them are in the hundreds of millions. There are many ways to improve things a little, and they are being tried. But to achieve a fundamental jump in our programming capacity, we need to rethink what we are doing from the beginning.

The problem

I believe that the problem lies in an obsolete fundamental view of programming and programming languages. A widely accepted view can be paraphrased:

The programmer's job is to design an algorithm (or a computation) for carrying out a desired task, and to write it down as a complete and precise set of instructions for a computer to follow. High level programming languages simplify the writing of these instructions by providing basic building blocks for stating instructions (both control and data structures) that are at a higher level of the logical structure of the algorithm than those of the basic machine.

This view has guided the development of many programming languages and systems. It served well in the early days of computing, but in today's computational environment, it is misleading and stultifying. It focusses attention on the wrong issues, and gives the most important aspects of programming a second-class status. It is obsolete in the same sense that binary arithmetic is obsolete -- the things it deals with are a necessary part of computing, but should play a subsidiary rather than central role in our understanding of what is relevant.

As computer technology (for both software and hardware) matures, our growing ability to create complex systems has led to two basic changes in the nature of programming. Each of them has been hindered by the traditional idea of programming languages.

1. *The building blocks out of which systems are built are not at the level of programming language constructs. They are complex "subsystems" or "packages," each of which is an integrated collection of data structures, programs, and protocols.*

By making use of existing modules, a programmer can deal with design at a much higher level, creating an integrated system with capacities far greater than a program that could be built with the same effort "from scratch." Components for general tasks (such as memory management, user interface, file management and network communication) can be designed once, rather than reconstructed for each system that needs the capability. Unfortunately, in current programming practice this is more of an ideal than a reality. The difficulties of using existing packages often make it easier to replicate their function than to integrate them into a system. The only way such packages are generally used is by building programs within an "operating system" that provides facilities within a uniform environment. Only those packages that are needed by the majority of users find their way into operating systems, and the facilities for using them are complex and ad hoc relative to modern programming languages.

2. *The main activity of programming is not in the origination of new independent programs, but in the integration, modification, and explanation of existing ones.*

The second change grows from the first. As we are able to build more complex programs, we develop systems that grow to fit an environment of needs, rather than carrying out a single well-specified task. A complex system (such as one for airline reservations, text preparation and formatting, or statistical analysis) evolves over many years, increasingly coming to fit the needs of those who use it, and adding new capacities as hardware advances make them practical. As additional needs and possibilities arise, it should be far easier to modify and combine existing well-tested systems than to build new ones. In most cases, the needs for continuity in the use of the system (including "upward compatibility" for existing data and user programs) make it impractical to start from scratch. Using current programming techniques, systems often reach a point where the accretion of changes makes their structure so baroque and opaque that further changes are impossible, and the performance of the system is irreversibly degraded. The situation is further complicated by the fact that modifications are often done not by the original builders, but by new programmers with an incomplete or even wrong understanding of the system.

The difficulties in building and modifying large systems have long been recognized and lamented. They have led to various schools of "structured programming," and to the emphasis on restriction and discipline in the design and use of programming languages. There is a large body of lore shared by practicing programmers, providing ways to recognize the problems and guidelines for avoiding the most obvious of them. These include bodies of standards and conventions designed to avoid misunderstanding and miscommunication. But ultimately the solution lies not in greater

discipline but in more adequate tools.

Towards a solution

Just as high-level languages enabled the programmer to escape from the intricacies of a machine's order code, *higher-level programming systems* can provide the means to understand and manipulate complex systems and components. In order to do this, we need to shift our attention away from the detailed specification of algorithms, towards the description of the properties of the packages and objects with which we build. A new generation of programming tools will be based on the philosophy that what we say in a programming system should be primarily *declarative*, not *imperative*:

The fundamental use of a programming system is not to provide a set of instructions for accomplishing a task (or carrying out an algorithm), but is to provide for the expression and manipulation of descriptions of computational processes and the objects on which they are carried out.

Current languages provide only scattered specialized mechanisms for description. Declarations are a ubiquitous form of low-level description, and assertions about the state of a computation have been proposed as a basis for automatic programming and verification. But if we look at what a programmer would say about a program to a colleague who wanted to work on it or use it, very little of the description appears anywhere in the "code." If (either because of idealism or coercion) the programmer has included comments, they can provide useful, but local, description. If further (almost always through coercion) the program has been documented, there may be more global descriptions. In large systems, documentation will include a careful specification of protocols and conventions not belonging to any one program, but vital to the system as a whole. These various pieces of description are scattered, and for the most part not usable except as English text to be read. There is an increasing amount of work on "specification languages" and "structured design" formalisms, which goes in the right direction, but these still play only a peripheral role in the normal activity of programming and debugging.

I want to turn the situation on its head. The main goal of a programming system should be to provide a uniform framework for the information that now appears (if at all) in the declarations, assertions, and documentation. The detailed specification of executable instructions is a secondary activity, and the language should not be distorted to emphasize it. The system should provide a set of tools for generating, manipulating, and integrating descriptions. The activity that we think of as "writing a program" is only one part of the overall activity that the system must support, and emphasis should be given to understanding, rather than creating programs.

The rest of this paper explores some of the consequences of this view, and makes some suggestions as to what a higher-level programming system might look like. It is an attempt to lay out the problems, not to solve them. It will take many years of research before these speculations can be backed up with concrete evidence.

A motivating example

One of the best ways to understand a general view of computing is to look at the examples used by those who hold it. Knuth's opus, *The Art of Computer Programming* [50], begins by discussing the Euclidean algorithm. In a clear and simple way it exemplifies the basic notions of algorithm and program from his viewpoint. The LISP 1.5 Manual [51] includes the LISP interpreter written in LISP, illustrating the manipulation of symbolic structures, and the ability to treat the programs themselves as symbolic data. The view proposed in this paper is best illustrated by an example at a very different level.

Imagine that you have come to work as a system designer and programmer for a large university. The task you are given is:

The current situation: *The university administration has a computer system for scheduling and planning room usage. Users at several sites on campus access the system through interactive graphics terminals that display building floorplans as well as text and other graphic data. There is a mini-computer running each cluster of terminals, connected to the central campus facility by a communication network. Each cluster has a device capable of printing out floorplans and graphic data like that displayed on the terminals. Large-scale data storage is at the central campus computer facility.*

The system keeps track of the scheduled usage of all rooms, including long-term lab and office assignments, regular class schedules, and special events. It is able to answer questions about current scheduled usage and availability. Querying is done from the terminals, through structured graphic interaction (menus, standardized forms, pointing, etc.) and a limited natural language interface. The system does not make complex abstract deductions, but can combine information in the data base to answer questions like "Is there a conference room for 40 people with a projection screen available near the education building from 3 to 5 on the 27th?". Users with appropriate authorization enter new information, including the scheduling of room use and changes to the facilities (including the interactive drawing of new or modified floorplans). In addition to the current assignments, the system keeps a history of usage for analysis. Standard statistical information and data representations (such as tables, bar charts, graphs, etc.) are produced on demand for use in long-range planning.

Your assignment: *The dean wants the system to provide more help in making up the quarterly classroom assignments. She wants to give it a description of the courses scheduled for a future quarter and have it generate a proposed room assignment for all courses. In deciding on assignments, the system should consider factors such as expected enrollment (using past data and whatever new information is available on estimated enrollments), the proximity of rooms to the departments and teachers involved, the preference for keeping the same location over time, and the nature of any special equipment needed. It should print out notices for each teacher, department, dean, and building supervisor, summarizing the relevant parts of the plan. Any of these people should be able to use the normal querying system to find out more about the plan, including the motivations for specific decisions. Properly authorized representatives of the dean's office should be able to request changes in the plan through an interactive dialog with the system, in which alternatives can be proposed and compared. When a change is made, the system should readjust whatever is necessary and produce new notifications for the people affected.*

A system like this is just at the edge of our programming powers today. It would take many programmer-years of effort to build, and would be successful only if the project were managed extraordinarily well, even by the standards of the most advanced programming laboratories. But it is not hard because of the intrinsic difficulty of the tasks the system must carry out. It combines hardware and software facilities that have been demonstrated in various combinations a number of times. Even the question-answering and assignment-proposing components are within the bounds of techniques now considered standard in artificial intelligence.

The problem is in our power to organize systems. The integration of all of the components of the "initial system" would be a major achievement, calling on our best design tools and methodologies. The idea that a new programmer could come in to such a system and make changes widespread enough to handle the "assignment" is enough to make an experienced programmer shudder. It would be hard enough to add new types of questions (such as explanations for decisions), new information (such as distances and estimated enrollments), and new output forms (such as schedule summaries for departments). But even more, we are trying to integrate a new kind of data (projected plans) into a system that was originally built to handle only a single current set of room assignments and a record of their history. These projected plans must be integrated well enough for all of the existing facilities (including floorplan drawing, question answering, statistics gathering, etc.) to operate on them just as they do with the initial data base.

In order for any of this to be feasible, we need a uniform language across all of the different components and data structures, in which the objects and processes of interest can be described. This description has to be at a conceptual level that makes it possible to ignore details of implementation whenever possible, and to state the interconnections between objects and processes of widely varying structure.

Three domains of description

A system of this complexity can be viewed in each of three different "domains," *subject*, *interaction*, and *implementation*. Each viewpoint is appropriate (and necessary) for understanding some aspects of the system and inappropriate for others. We will look at the example from each of these viewpoints in turn, then discuss how they might be embodied in a programming system.

The subject domain. This system, like every practical system, is about some subject. There is a world of rooms and classes, times and schedules, that exists completely apart from the computer system that talks about them. The room or the course can't be in the computer -- only a description of it. One of the primary tasks in programming is to develop a set of descriptions that are adequate for talking about the objects being represented. There are descriptions for things we think of as objects (e.g., buildings, rooms, courses, departments) and also for processes (e.g., the scheduling of events). These descriptions are relative to the goals of the system as a whole, and embody a basic view of the problem. For example, what it takes to represent a room would be fundamentally different for this system and for a system used by contractors in building construction.

All too often, the development of descriptions in this domain is confused with the specification of data structures (which are in the domain of implementation). In deciding whether we want a course to be associated with a single teacher, or to leave open the potential representation of team-teaching, we are not making a data structure decision. The association of a teacher (or teachers) with a course may be represented in many different data structures in many different components of the system. One of the most common problems in integrating systems is that the components are based on different decisions in the subject domain, and therefore there is no effective way to translate the data

structures. A programming system needs to provide a powerful set of mechanisms for building up and maintaining "world views" -- coherent sets of description structures in the subject domain that are independent of any implementation. Each component can then implement part or all of this in a way that will be consistent with both the structure of that component and the assumptions made in other components.

A complex system like that described above will need hundreds of different categories of objects. Some of these (such as classrooms and courses) will be unique to the system. Others (such as times and dates, schedules, physical layouts) will be shared across a wide range of systems. They will be related into hierarchies of abstraction. We can think of a *seminar*, a *course*, and a *special lecture* as examples of a more general class of *event*, all having a time, a place, etc. For some purposes, this is the right level of generality. For other purposes, we need to distinguish carefully and use special information associated with each. A major part of building up systems will be the systematic development of descriptions that provide a uniform medium for describing components and their interactions.

The domain of interaction. This system, like every functioning system, can be viewed as carrying on an interaction with its environment. As we will discuss in a moment, the choice of "system" and "environment" is relative to a specific viewpoint, but for the moment let us consider the system as viewed by the users. In this domain, the relevant objects are those that take part in the system's interactions: users, files, questions and answers, forms, maps, statistical summaries and notifications to departments. The processes to be described are those like querying the system, describing a new event to be scheduled, and proposing a schedule for a new quarter.

The domain of interaction is concerned with descriptions that are largely orthogonal to those in the subject domain. We can talk about a question as having certain characteristics (e.g., looking for a yes-no answer) independently of whether it is about a room or a lecture. We can talk about the filling out of a form without reference to its specific contents. It is also (and more importantly) independent of the domain of implementation. From the traditional viewpoint, this independence is a bit more difficult to see than the independence of interaction and subject matter. Whereas a subject domain object (like classroom) clearly cuts across large parts of the system, an interaction object (like a question or the process of filling out a form) is typically handled by a single component and described in terms of its implementation. But for the same reasons we want to keep subject domain descriptions independent of their implementations, we must do the same with interaction descriptions.

This view can be applied recursively to sub-parts and components of the system. In looking at one part (such as the on-site processor and its cluster of terminals) we can view it as an independent system interacting with an environment including both the users and the other parts of the overall system, such as the centralized data store. Even within a single implementation module (e.g., the question-answerer), we often want to describe what is happening as an interaction between several conceptual sub-systems (the parser, the semantic analyzer, etc.). As with the larger system, it is vital to keep in mind the distinction between the interaction domain and the implementation domain. In order for two conceptual subsystems to be described as interacting, they need not be implemented on physically different machines, or even in different pieces of the code. In general, any one viewpoint of a component includes a specification of a boundary. Behavior across the boundary is seen in the domain of interactions, and behavior within the boundary is in the domain of implementation. That implementation can in turn be viewed as interaction between sub-components.

It is in the domain of interaction where there is currently the most to be gained from developing bodies of descriptive structures to be shared by system builders. There are already many pieces that can be incorporated, including protocols (e.g., network communication protocols, graphics representation conventions), standardized interaction facilities (e.g., ASKUSER and DLISP in Interlisp [52, 53], the Smalltalk display programs [31]), front-end query packages (in various artificial intelligence programs), data-base standards, and so forth. Currently each of these is in a world and formalism of its own. Given a sufficiently flexible tool for describing and integrating interaction packages, this level of description will be one of the basic building blocks for all systems.

The domain of implementation. Every computer system operates on a set of physical devices with hard-wired mechanisms for for storing and manipulating data. It is in this domain that we normally think of programming. The detailed choice of algorithms, data structures, and configuration is determined by properties of the hardware and of the descriptive languages we have available for specifying its behavior. However, it would be a mistake to equate this domain with our current notions of programming. The objects in this domain include everything from individual memory bits and subroutines to subsystems (e.g., the file system, the memory management system, the operating system), running processes, hardware devices, and code segments. They include those things we talk about in programs, those we talk about in the debugger, and those found in the machine and hardware manuals. In this domain, as in the others, a uniform system for description is needed, which is not primarily a language for specifying a set of instructions.

In addition to all those things that are directly derivable from the code, the manuals, and the state of the machine, there is also a body of process description. For example, it may be a property of a specific memory-management package that it periodically undergoes a garbage collection period of up to 5 seconds during which no new memory allocations can be made. Such "performance" characteristics may be vital for understanding the interactions of a component with the rest of the system, but are not explicit in its code. Other descriptions bring into focus things that may be implicit in the code. A file system may delete a file if its creation process is interrupted in certain ways that would leave it in danger of being inconsistent. The code that does this may be distributed through various checks and actions, but for the programmer attempting to understand the program it is necessary to have a coherent overview of what is happening.

Similarly, many of the things we think of as properties of data structures are actually conventions spread through the code that manipulates them. Much of the work in structured programming has been to isolate these conventions into access functions that go into a "module" with the data structures [29, 32, 34]. The object-oriented style of programming encouraged by Smalltalk [31] is another attempt to provide this kind of modularity. A system for implementation description would provide for stating these in a more general way along with those things that now appear only in the comments. As with procedures, data structures can also have implicit properties (e.g. the expected maximal size of variable length fields) that need to be stated explicitly in order for a person to understand how they will interact with other components.

The boundary between hardware and software has been blurred in the past few years through developments such as micro-code, uniform logic arrays, and the extensive use of virtual machine architecture. A programming system based on description pushes this one step further. The emphasis is on an overall description of a component rather than the instructions needed to cause some piece of hardware to run it. A piece of software and a piece of hardware designed to achieve the same purpose would have descriptions that differed in details (e.g. timing), and in the specific aspect that described the code (or logic circuits) used to carry out the steps. They might be identical in the domain of interaction, and even to a large degree in the domain of implementation (for

example in the logical description of their data structures).

A sketch of a higher level programming system

So far, we have been talking in a general way about the different domains of description and the kinds of things that might be said about a component or system from each of their viewpoints. This doesn't yet provide a coherent picture of what a program will be. What do we see on the printed page or on the screen? How is it organized? How do we do anything with it?

Once again, it is important not to let our preconceptions get in the way. Notions such as code, listing, file, and compilation are based on the idea of a program as a set of instructions. There need not be any directly corresponding objects in a higher-level system. Instead, it should be based on something much more like what we now think of as an artificial intelligence system, with its "knowledge base" of assertions and procedures. There will be a set of interrelated descriptions, stored in a form that makes it possible to retrieve, manipulate and display them. These will include *prototypes* for categories of objects and processes (like *classroom*, and *filling out a form*), and *instances*, which correspond to individual objects and processes in one of the domains (such as the course CS 365 in Winter 1978, the contents of the third page of next quarter's schedule, and the process currently running in the data-base server). Instances can be described by more than one prototype, and prototypes are related into hierarchies with different degrees of abstraction. The details of all this are still a matter for extensive research. One set of possibilities is being explored in KRL [2,3], but the basic idea of higher level systems could be implemented using other descriptive representations, such as semantic networks [4, 6, 10, 11, 12] and other frame based systems [5, 7].

In addition to the basic description system, there will be a wide range of commonly useful prototypes and instances. Some of them (e.g., graphics formats, dates and times, communication protocols) will be in the subject and interaction domains. Others (such as the data structures used in a particular data base) will be in the implementation domain. Some (such as descriptions of specific pieces of hardware and software that are being used) will be specific to the programming system. Others (such as those for abstract objects like sets and sequences, and for process structures) will be very general. In approaching a problem, a programmer will make use of this vocabulary of concepts and descriptive categories, both for interfacing with existing components and organizing new ones.

A programmer's use of such a system will be highly interactive. Since the understanding of a component comes from having multiple viewpoints, no single organization of the information on a printed page will be adequate. The programmer needs to be able to dynamically reorganize the information, looking from one view and then another, going from great generality down to specific detail, and maneuvering around in the space of descriptions to view the interconnections. This will require an interface that is more sophisticated than the question-answering interfaces now used on artificial intelligence knowledge bases. It seems likely that pictorial representations, interactive graphics, and "intelligent summarizing" will play a large role.

Of course, there always remains the task of providing a description of each component that is detailed enough to allow the system to run it. This will be part of providing a broader description, and may be done in stages. A very abstract specification of what a component does will be sufficient for a kind of "high-level debugging" in which its interactions with other components can be analyzed and described without carrying out the steps at the the lowest level. There is a whole range of operations that are now thought of as "automatic programming", which will enable the programmer to let the system fill in details once the overall behavior of the component has been specified. Some of this will be based on standardized defaults, others on automated analysis of

performance characteristics. It will all be based on the availability of descriptions in the implementation domain of the various machines and subsystems being used.

As systems become more complex, the level of desirable invisibility will rise. Current high-level programming languages do not give the user the opportunity to decide just how the hardware registers of the machine will be used to store variables, since there is much more to be gained by a uniform integrated approach to their use by the compiler. Similarly, much of what we now think of as data structure and algorithm specification will be handled by programs that can take into account much more complex efficiency considerations than are practical for a human programmer.

In summary, a programmer will use a programming system that contains a base of knowledge about the system he or she is working on, and of other potential components and concepts of programming that might be of use. The programmer will modify the descriptions of existing components, and create high-level descriptions of new ones to be added. These modifications and additions may be from the viewpoints of all of the different domains, and will be done in cooperation with the system, which checks for consistency, looks for consequences of new information, etc. Checking and debugging will be done at a variety of levels as the description becomes more detailed. The system will attempt to fill in details that are needed to completely specify the implementation, so the system as a whole can be run. The debugging process will make use of sophisticated reasoning processes that can use all of the different domains of description in analyzing and reporting what is happening.

What needs to be done

To create a system like the one just described, we need further research in three distinct but interrelated areas: the development of an effective descriptive calculus; the creation of a body of descriptive concepts for computational processes; and the building of a complex integrated system which uses them.

A descriptive calculus. The main thrust of these ideas depends on the ability to create and manipulate descriptions in an effective, understandable way. There are existing formalisms for description (for example, the predicate calculus) which are clear and well-understood, but lack the richness which makes natural description effective. They can be used as a universal basis for description but only in the same sense that a Turing machine can express any computation. They lack the higher-level structuring which makes it possible to manipulate descriptions at an appropriate level of conceptual detail.

The requirements for a descriptive language of the kind I propose are quite different from those used in the mathematical foundations of computation, or for program verification. Work in these areas (see, for example the collection of papers in [15]) emphasizes the use of descriptive languages in rigorous proofs of the properties of programs. A higher-level programming system must instead emphasize the use of descriptive languages for communication. The concentration must be on those aspects which aid in giving a person a clear overall understanding (at variable levels of detail and from multiple points of view), rather than on those aspects which increase the mathematical tractability of the descriptions. There are artificial intelligence formalisms (such as semantic networks [4, 6, 10, 11, 12] and KRL [2,3]) with increased dimensions of expressiveness, but are not yet at a level of precision which would make them sufficiently understandable to be used in a system of the required complexity. The characteristics which they explore (and which will need to be part of a formalism to be used in a higher-level programming system) include:

Prototype hierarchies. The nouns and verbs of a natural language can be organized into hierarchies (or taxonomies) which capture much of the logical structure of what they describe. We know that a *dog* is an *animal*, and the answers to questions about dogs will often be derived through general properties of animals. Systems such as semantic networks treat these deductions specially, rather than dealing with them uniformly as a set of universally quantified implications. This leads to greater efficiency for the common calculations, and provides a structure which makes it much easier for a programmer to organize a knowledge base. These hierarchies contain information which could be thought of as a set of independent facts, but has additional structure in the same sense that a structured program structures a set of steps and jumps.

The centrality of defaults. Most logical calculi are optimized for handling generalizations which are either true or false. They do not provide means for stating generalizations that are not completely universal, but are "usual" or "normal" or "expected". In natural descriptions of any kind, people draw heavily on the ability to use information of a less formal sort, and to use a kind of *ceteris paribus* reasoning in which a standard fact is assumed true unless there is an explicit reason to believe the contrary. One of the major directions in artificial intelligence representation research is the attempt to provide this capability in a formally clear system. The notion of a *default* value is familiar to every programmer, but its place in a formal calculus needs to be carefully worked out.

The suppression of exceptional details. One of the major reasons for using precise formalizations is that they make everything explicit. For some purposes this is good, but there are times when understanding can come only through the suppression of detail. If we are trying to formally describe a program which has a normally simple input-output behavior (e.g. one that copies data from one place to another) we want to describe that behavior in a way which highlights that simplicity. If there are exceptional cases (e.g. when the storage allocator fails to find a sufficient block), these need to be described, but in a secondary place. The basic description of an object cannot be cluttered up with all of the details needed for handling the contingencies. Formalisms used in denotational semantics for programs demonstrates this problem well. In order to deal with a special escape at all, they demand that even the simplest programs be described as operating on continuations, environments, etc. and this description permeates every level of what is said.

Multiple levels of abstractions and instances. In dealing with programs and processes, we run into complexities involving the *instantiation* of general classes. For example, a specific algorithm (such as Euclid's algorithm) can be thought of as an instance of a general class (numerical algorithms), or as a class whose instances are programs implementing the algorithm. Each of those programs is in turn both an instance (of the class of formal objects known as programs) and a description of a class of process instances, each of which is carrying it out. If we look at the finer structure of programs, such as the instantiation of variables or pieces of code within loops, similar phenomena arise. Higher-order and typed logics deal in certain ways with the notion of a class (predicate) which is also an instance, but their austerity makes them inadequate for capturing the rich set of ways in which people interleave levels of abstraction. Artificial intelligence formalisms have not yet dealt adequately with these issues, which are currently a topic of active investigation.

A basis for describing processes. Given a formalism for descriptions in general, we need a body of prototypes for describing those things which are common to all of our programs (e.g., processes, programs, data structures, communication acts). This is a necessary kind of *library*, just as a library of standard data-structures and statistical routines is a necessary part of a system for statistical manipulations. It need not be fixed once and for all, but a good deal of it must be in place before the system is usable, and it must be held relatively uniform if the system is to be extendable.

In this area, there are many ideas floating around that need to be captured in a more precise form. The success of higher-level programming systems will depend on having a coherent body of descriptive categories which can capture all of their variety.

Modularity and structured procedures. There has been a good deal of attention in recent years to the higher-level structure of control constructs. In addition, languages based on data abstractions (such as CLU [32], Alphard [34], and Mesa [29]) provide for larger *modules* which encapsulate collections of data structures and procedures. Beginning from a different point of view, *structured system description languages* [16, 17] provide a body of conceptual tools for describing the overall structure of large systems. We need a consistent way of talking about modularization and interaction between semi-independent modules which can be applied to system structure at all different levels of detail.

Structured data objects. Current work on programming language constructs often emphasizes the structure of the sequence of operations, in terms of loops, recursive calls, etc [14]. A related notion in describing processes is the ability to hide detail by allowing the combination of objects into a larger "structured object", and to define unitary operations on this higher level object which invoke collections of operations on the components. This has been explored for simple mathematical objects (e.g. lists in LISP's MAP functions [51], vectors and arrays in APL [20], sets in SETL [26] and VERS [19]), and seems applicable to more specialized semantic objects (in all of the three domains) as well. In many cases, much of what is now thought of as control structure can be implicit in the data structure, leading to notions of "nonprocedural" or "procedureless" languages [21, 22, 24]. The interaction between control and data structure needs to be put into a theoretical framework.

Program factoring -- objects and procedures. In viewing a process as a structured sequence of individual steps, there are different ways to think about what each of those steps is. Most programming languages lead the programmer to think in terms of *operations* (either primitive or programmer-defined) to be carried out on *arguments*. Some (such as Simula [27], Smalltalk [31] and Plasma [47]) think of typed *objects* which receive and interpret *messages*. Instead of organizing code around a single procedure (e.g. *print* or *plus*) which selects its action according to data type, they define *classes* (such as integer, list, etc.) which select what to do on the basis of the message. Artificial intelligence languages take a more general approach in using *pattern directed invocation* [1, 28]. Each step specifies a pattern (or *goal*) to be achieved. A data base of *pattern-action pairs* is accessed to decide what steps to carry out. Each of these viewpoints is best for some aspects of programming, and we need to understand how to integrate them into a larger framework.

States and transitions. There are two complementary ways of looking at a computational process -- as a sequence of operations, or as a sequence of states. This duality has been exploited in the mathematical theory of computation, but has not really been integrated into programming languages. Transition nets and Petri nets [38, 40] are state-oriented, rather than operation-oriented. Production systems [43] are state-oriented, with each production specifying a

partial state description, and an appropriate transition function (not the name of the new state, but a set of operations which produce the new state). Languages which provide ways of specifying actions to be taken on special conditions [37] are really mixing state-transition description with the normal operation sequence. As with the operation/object distinction above, the goal is to find a synthesis which allows a process to be described using a mixture of the conceptual viewpoints, and to be run on the basis of that description.

Interacting processes and communication. The notions above deal primarily with a single process. The most significant direction in computing over the coming years will be towards multiple processes, both virtual (e.g. organizing a speech system as a series of separate processes, even if it runs on a single PDP-10 [46, 49]) and actual (e.g. networks of computers cooperating on a single task). There are a number of issues which have been dealt with by system designers at lower levels (like operating systems) which have not found their way into higher level languages. There is also a wealth of metaphors provided by thinking of a computation as being carried on by a collection of independent individuals which must communicate by exchanging messages in a common language. We can draw many analogies from human communication -- What language do they talk? Which subsystems need to be multi-lingual? What are the discourse rules for establishing and controlling message flow? Is it possible to learn a second language? How can two processes make use of shared knowledge to increase the efficiency of communication? How can one process make use of an internal model of another process, in order to facilitate communication and cooperation?

A complex system. The kind of higher level programming system described here is itself a massive and complex system. Its subject matter is not an external one, like room-scheduling, but the reflective one -- the subject of programming itself. There is a bootstrapping problem. The system needed to realize these ideas can be built only with a set of tools that help in the construction of large integrated systems. Perhaps our current systems are good enough to start the bootstrapping, but that first step will be a big one. It will take a good deal of careful system-organization research before something of this scale can be effectively constructed.

Conclusion

The title of this paper is consciously provocative. Rather than asking how we might improve programming languages, I argue that we should look at things from a completely different vantage point in which there is little concern with specifying programs as we now see them. It grew out of work on description systems which emphasize concepts such as multiple viewpoints for description, default knowledge, and prototypes. These ideas can be applied to make the rich set of programming concepts represented in the literature more applicable to the real practice of programming. In order to make significant progress, we need to deal with the problems of "programming in the large". Once we begin to deal with networks of independent processors, it will become even more important to deal explicitly with global properties that cannot be understood on the basis of individual program instructions.

I believe that the the type of higher level programming system described here is a step in the right direction. There is no way to demonstrate this with certainty. As stated at the beginning, this paper is not a presentation of a worked-out solution. All we have at this point is an intuition that a particular way of approaching the problem will lead to new insights and results. The paper is intended to provoke thinking and suggest some new directions. If they turn out to be fruitful directions, we have years of hard and exciting work ahead.

Acknowledgements

In a paper of this kind, it is impossible to properly credit the sources of the ideas. It grew out of ongoing interactions with people who hold very similar ideas in different forms, and it is really just an expression of the current state of my intellectual environment. My joint work with Danny Bobrow and Brian Smith on the theoretical foundations of KRL has been a primary source of ideas, and the rest of the Stanford/Xerox KRL research group (David Levy, Mitch Model, Don Norman and Henry Thompson) have been involved in all stages of our thinking. Stanford computer science students in the CS365 seminar in 1977 pushed and probed on many of the ideas about procedures, which in turn came from the authors of the papers we read there (included in the list of references). The Xerox PARC environment has been a context in which the problems of "programming in the large" are well understood, and has provided a wealth of ideas and examples, including the work of Alan Kay and his group on Smalltalk, the implementation of the Mesa programming language, the development of programming environments by Warren Teitelman and Larry Masinter, Bob Sproull's understanding of graphics systems and protocols, and Peter Deutsch's views on system organization and programming environments. In addition, the cybernetic notions of Humberto Maturana as introduced to me by Fernando Flores have led to subtle but very important shifts of perspective in the way I look at systems of all kinds. I am also grateful to Allen Perlis, Peter Deutsch and Jim Horning for extensive and insightful comments on an earlier draft of this paper.

References

Note for the draft version: This reference list is still somewhat rough. It contains all the references from the Stanford seminar mentioned in the last section, along with others mentioned in this paper. It may be better to make it more selective, so I have not bothered to clean it up totally, but will wait for comments first. My idea in including it was that it gives a broad picture of the kinds of issues which need to be taken into account. I am a little worried that it is so broad as to be overwhelming. It is also somewhat idiosyncratic, since it consists of papers that I was familiar with. I welcome suggestions of additional papers or substitutes on all of the topics.

Description Formalisms

- [1] Daniel Bobrow and Bertram Raphael, "New programming languages for AI Research", *Computing Surveys* 6:3 (September 1974).
- [2] Daniel Bobrow and Terry Winograd, An Overview of KRL, a Knowledge Representation Language, *Cognitive Science* 1:1, January 1977, pp. 3-46.
- [3] Daniel Bobrow, Terry Winograd, and the KRL research group, Experience with KRL-0: One cycle of a Knowledge Representation Language, *Fifth International Joint Conference on Artificial Intelligence*, pp. 223-227.
- [4] Ron Brachman, What's in a concept: Structural foundations for semantic networks, *Int. J. Man-Machine Studies* (1977) 9, pp. 127-152.
- [5] Randy Davis, Knowledge about representations as a basis for system construction and maintenance, in *Pattern Directed Inference Systems*, Academic Press (in press).

- [6] Richard Fikes and Gary Hendrix, A network-based knowledge representation and its natural deduction system, *Fifth International Joint Conference on Artificial Intelligence*, pp. 235-246.
- [7] Ira Goldstein and Bruce Roberts, Nudge, a knowledge-based scheduling program, MIT AI-MEMO 405 (February 1977).
- [8] Pat Hayes, Some problems and non-problems in representation theory, AISB conference, 1974, pp. 63-79.
- [9] Pat Hayes, In Defence of Logic, *Fifth International Joint Conference on Artificial Intelligence*, pp. 559-565.
- [10] Hector Levesque, A procedural approach to semantic networks, TR-105 Dept. of Computer Science, U. of Toronto, 1977.
- [11] David Rumelhart and Donald Norman, The Active Structural Network (Chapter 2) from *Explorations in Cognition*, 1975.
- [12] Bill Woods, "What's in a link?", in Bobrow and Collins (eds.) *Representation and Understanding*, 1975.

Formalisms for specifying programs

- [13] Burstall, R.M. and Goguen, J.A., Putting Specifications Together, SIJCAI, 1977.
- [14] Edsger Dijkstra, *A Discipline of Programming*, Prentice Hall, 1976.
- [15] E.J. Neuhold (ed.), *Formal Description of Programming Languages*, North-Holland, 1978.
- [16] Kenneth T. Orr, *Structured Systems Development*, Yourdon Press, 1978
- [17] Douglas Ross, Structured Analysis Design Technique, ????
- [18] R.D. Tennent, "The Denotational Semantics of Programming Languages", *CACM*, August 1976, pp. 437-453.

Structured objects and structured procedures

- [19] Jay Earley, High Level Operations in Automatic Programming, SIGPLAN notices 9:4 (1974), pp. 34-42.
- [20] A.D. Falkoff and K.E. Iverson, The design of APL, *IBM Journal of Research and Development*, 1973, pp. 324-334.
- [21] Clair Goldsmith, The design of a procedureless programming language, SIGPLAN notices 9:4 (1974), pp. 13-24.

- [22] Hammer, Howe, and Wladawsky, an Interactive Business Definition System SIGPLAN notices 9:4 (1974), pp. 25-33.
- [23] Robert Kowalski, Predicate Calculus as a programming language, IFIP proceedings, 1975.
- [24] Burt Leavenworth and Jean Sammet, An Overview of nonprocedural languages, SIGPLAN notices 9:4 (1974), pp. 1-12.
- [25] John Reynolds, GEDANKEN--A Simple typeless language based on the principles of completeness and the reference concept, CACM 13:5 (May 1970), pp. 308-319.
- [26] Jacob Schwartz, On Programming: an interim report on the SETL project; Installment I: Generalities, NYU Courant Institute, February 1973.

Program factoring -- modules, objects, and procedures

- [27] Birtwistle, Dahl, Myhrhaug and Nygaard, SIMULA BEGIN, 1973.
- [28] Randy Davis, Generalized procedure calling and content directed invocation, *Proc. ACM Conference on AI and Programming Languages*, August 1977.
- [29] Charles M. Geschke, James H. Morris Jr., and Edwin H. Satterthwaite, Early Experience with Mesa, *CACM* 20:8 (August 1977), pp. 540-552.
- [30] Chuck Geschke and Jim Mitchell, *On the problem of uniform references to data structures*, Xerox PARC CSL-75-1, January 1975.
- [31] Adele Goldberg and Alan Kay, *Smalltalk-72 Instruction Manual*, Xerox PARC SSL 76-6, 1976.
- [32] Barbara Liskov, Alan Snyder, Russell Atkinson, and Craig Schaffert, Abstraction Mechanisms in CLU, *CACM* 20:8 (August 1977), pp. 564-576.
- [33] Vaughn Pratt, The Competence/Performance dichotomy in programming, 4th ACM symposium on the principles of programming languages, 1977, pp. 194-200.
- [34] Mary Shaw, William A. Wulf, and Ralph L. London, Abstraction and verification in Alphard: Defining and specifying iteration and generators, *CACM* 20:8 (August 1977), pp. 553-562.
- [35] Guy Steele, LAMBDA, the ultimate imperative, MIT-AI Memo 353, March 1976.
- [36] Guy Steele, LAMBDA, the ultimate declarative, MIT-AI Memo 379, November 1976.

States and transitions

- [37] John Goodenough, Exception Handling: Issues and a proposed notation, *CACM* 18:12, December 1975, pp. 683-696.

- [38] Anatol Holt, Introduction to Occurrence Systems, in Jacks (ed.) *Associative Information Techniques*, Elsevier 1971. pp. 175-203.
- [39] E. Humby, *Programs from Decision Tables*, McDonald/Elsevier, 1973.
- [40] P.E. Lauer and R.H. Campbell, A Description of path expressions by Petri Nets, Second ACM symposium on principles of programming languages, 1975, pp. 95-105.
- [41] Howard Lee Morgan, Event Sequenced Programming, Cornell Dept. of operations research Tech report 119 (July 1970).
- [42] Chuck Reiger, "The Commonsense algorithm as a Basis for Computer Models of Human Memory, Inference, Belief and Contextual Language Comprehension", Schank and Nash-Webber (eds.), *Theoretical Issues in Natural Language Processing*, 1976. pp. 180-195.
- [43] Mike Rychener, Production Systems: a case for simplicity in AI Control Structures, draft of paper submitted to ACM National Conference, 1977.
- [44] Earl Sacerdoti, The non-linear nature of plans, 4IJCAI, pp. 206-214.
- [45] Michael Zisman, A Representation for Office Processes, Wharton Department of Decision Sciences Working paper 76-10-03.

Interacting processes and communication

- [46] Jeff Barnett, Module linkage and communication in large systems, in D.R. Reddy (ed.) *Speech Recognition*, pp. 500-520.
- [47] Carl Hewitt, Viewing control structures as patterns of passing messages, MIT AI Memo 410 Dec. 1976.
- [48] Lampson, Mitchell and Satterthwaite, On the transfer of control between processes, Proceedings of Programming Symposium, Paris, April 1974 (#19 in Lecture notes in Computer Science, Springer Verlag, 1974), pp. 181-203.
- [49] Victor Lesser, Parallel processing in speech understanding systems: A Survey of design problems, in Reddy (ed.) *Speech Recognition*, 1975, pp. 481-499.

Other references

- [50] Donald Knuth, *The Art of Computer Programming*, Vol 1, *Fundamental Algorithms*, Addison Wesley, 1968.
- [51] M. Levin, et. al., *The Lisp 1.5 Programmer's Manual*, MIT, 1965.
- [52] Warren Teitelman, A display oriented programmer's assistant, *Fifth International Joint Conference on Artificial Intelligence*, 1977, pp. 905-915

[53] Warren Teitelman, et. al., *Interlisp Reference Manual*, Xerox PARC, 1975.

XEROX
BUSINESS SYSTEMS
Systems Development Division

To	Software Distributers	Date	June 12, 1978
From	Bruce Malasky	Location	El-Segundo
Subject	Brownie 1.0	Organization	SDD/SD

A Program for Maintaining Consistent Distribution Directories

Filed on: [Ifs-2]<malasky>memos>Brownie.memo

Overview

This memo describes a program for maintaining public *distribution* directories on a file server. As the number of users has increased and the slow speed communication lines have become more heavily utilized, it has become very important to provide easy access to *public* software. To this end many local IFS's now have their own versions of important directories such as <ALTO>, <ALTOFONTS> and <MESA>.

The Mesa program Brownie provides a method for semi-automatically making sure these directories are copies of the original distribution directories. It is hoped that software distributors, as well as IFS Administrators will make use of Brownie to control distribution of large systems. Brownie can be found on [Ifs-2]<Malasky>Brownie>Brownie.Image. For a while, I recommend that you run it with a disk that has the Mesa debugger installed. Since files being transferred will reside on the local disk (in Brownie.buffer\$), it is important to have a disk with about 2000 free pages.

Operation

Brownie is started in the same manner as any image file. NOTE: Brownie is currently a Mesa 3.0 program. Be sure to use RunMesa.run from <OLDMESA>. During initialization, Brownie looks for a file called Brownie.data on the local disk that describes the operations to be performed. Appendix A contains a sample data file and an explanation of its format.

How It Works

Brownie works by first enumerating files on both the source and destination hosts. Since the IFS software does not distinguish between empty and non-existent directories, there is a requirement for at least one file in the lowest subdirectory specified. In the example on the next page, if Brownie was asked to update [Host1]<mumble>foo>baz> from [Host2]<mumble>foo>baz>, the update would fail unless there was a placeholder file in [Host1]<mumble>foo>baz>. This requirement may change in the future. After the enumerations, Brownie then compares the write dates of files with the same name to determine which files require transferring. Files on the [source host] but not

the [destination host] will be transferred. If there are communication problems, Brownie automatically reopens timed out connections.

Filenames are compared without version numbers and with case ignored. However, any subdirectories not specified in the command line will be used as part of the name for comparison. When updating [Host1]<mumble>foo> from [Host2]<mumble>foo>, the file B.mesa will be transferred only if it's newer on Host2. The file a.mesa will be transferred regardless of its write date because it is not on the Host1. This will result in one version of a.mesa in the Foo>Bar> subdirectory and one in the Foo> subdirectory.

[Host1]	[Host2]
<Mumble>Foo>	<Mumble>Foo>
Bar>	a.mesa!2
A.mesa!1	Bas>
Bas>	B.mesa!3
B.mesa!4	Baz>
Baz>	a.bcd!1
	b.bcd!2

Planned Enhancements

In the future, Brownie will permit you to add your own module that will be able to fiddle with the files as they are transferred. This feature was motivated by the desire to alter command files automatically to use the new host.

The use of the local disk for buffering files will probably be removed at the time Brownie is converted to Mesa 4.0.

Brownie will optionally rename files on the same host. This will result in an enormous performance enhancement for rearranging directories on a single host.

Problem Reporting

All problems should be reported to Bruce <Malasky> (823-1469) or using sndmsg.

Appendix A

Below is a sample brownie.data and an explanation of its format:

```
{brief}
// logins follow
[Ifs-2]<Malasky>(secret)
[Iris]<guest>(guest)
[Maxc]<Malasky>(secret)
// These are commands
[Ifs-2]<Mesa> (mumble) ← [Maxc]<Mesa> ()
[Ifs-2]<MesaPup>Feb02> () ← [Iris]<MesaPup>Feb02> ()
[Ifs-2]<MesaLib>30> (bar) ← [Maxc]<MesaLib> (foo)
```

The first line is the mode it will run in. The mode is one of the following: {error, brief, verbose, debug}. The mode controls the amount of information that will be put into the mesa.typescript while Brownie is running. In debug mode it will print out passwords as well as other internal information. It is recommended that you run in brief mode.

The second line is a comment, but the leading // is required.

The next lines are used by Brownie to log into the various hosts. A host is delimited by [], a user name by <> and a password by (). Spaces are not allowed between fields of a login line. Logins are terminated by a second comment line.

All the remaining lines in the data file are command lines specifying what updates to perform. Each command line is in this format:

```
[Destination]<directory>SP(connect password)SP+SP[Source]<directory>SP(connect password)CR
```

where *SP* is a space and *CR* is a carriage return. No *'s are permitted in any of the directory specifications. Subdirectories for <To directory> and <From directory> are permitted, but they are ignored by Maxc. The connect password is required only when the login name does not have the necessary privilege. Nonetheless, the parentheses are required. Normally, a connect password will be specified for the [destination host] to store files, while one is usually not necessary to retrieve files.

For Xerox Internal Use Only -- June 21, 1978

EtherRcvr

June 21, 1978

1

Ethernet Receiver Exerciser

Diagnostic programs (such as MadTest, DiEx, TriEx, and TFU) often wish to run as many other tasks as possible to provoke failures caused by inter-task interference. This package runs the Ethernet receiver in promiscuous mode and copies every packet it hears into an internal buffer. The package consists of one file, EtherRcvr.br with one external procedure:

EtherRcvr(on) = true or false

If 'on' is true the Ethernet receiver is setup to receive every packet on the Ether. It returns true if the receiver was not on and false if a previous call to EtherRcvr has already started the receiver. If 'on' is false the receiver is shut down. It returns true if the receiver was on and false if it was already off. Packets are read into an internal buffer and discarded. Note that it is harmless to turn the receiver on when it is already on, or off when it is already off. To minimize overhead, EtherRcvr is written in Nova assembly language and uses interrupts. The static etherStatVec points to a 4 word statistics vector with the following format:

structure ESV:

```
[
good word 2      //# of packets rcvd with good status
bad word 2      //# of packets rcvd with bad status
]
```

WRC ALTO DOCUMENTATION

Program Name: FIRST

Description: Experimental Subsystem

Author: Bob Dattola

Date: May 30, 1978

Location: <dattola>first.dm on WRC IFS

FIRST is a document retrieval system based on the SMART system developed by G. Salton of Cornell University. It accepts natural, English language queries (sentences, phrases, or words), and passes the words through a stemming algorithm that ignores STOP words (very common non-content bearing words) and reduces suffix variations of the remaining words. The query is then searched against a data base of abstracts by computing a similarity function between the query and abstracts. Abstracts containing at least one common stem with the query are returned to the user in decreasing order of score, as determined by the similarity function. Thus, those documents most similar to the query (and hopefully most relevant) are output first. For more information on FIRST, see "FIRST -- Flexible Information Retrieval System for Text," Xerox Internal Report No. X76-00221, Dec. 1975 by R.T.Dattola.

The file **first.dm** is a dump file containing all the necessary files for creating and searching a FIRST data base. It consists of the three programs **FirstInit** **DocLoad** and **First**, the two files **STOP.first** and **Suffix.first**, and a copy of this documentation **first.doc**.

These programs are still considered to be experimental, so they are not being officially released as Alto Subsystems. Any problems and/or suggestions should be reported to me on Maxc account ISA.

Creating a FIRST Data Base

A FIRST data base consists of the following four files:

DocAbs.first -- Text of abstracts used for printing.

DocMat.first -- Weighted numeric vectors (one for each abstract) used for searching against query.

BFAtree.first -- A B-tree used for storing all the unique content bearing stems in the data base.

Auxtree.first -- A B-tree used for storing STOP words and suffixes.

These files are automatically initialized by running the program `FirstInit`. This program prompts the user for the following information:

- Data base name -- One line of text which is displayed when searching.
- Maximum document number -- The largest total number of documents to ever be stored in the data base.
- STOP word file -- The name of a file containing all the words to be treated as STOP words. The words need not be in any special order, but they must be separated by spaces or carriage returns. A standard STOP word list `STOP.first` is provided in `first.dm`.
- Suffix file -- The name of a file containing all the English suffixes to be used by the stemming algorithm. Same format as for STOP word file. A standard suffix list `Suffix.first` is provided in `first.dm`.

After running `FirstInit`, documents may be added to the data base by running `DocLoad`. At this time, input documents must conform to the following format requirements:

- Date, if present, must be the first string of characters in the document (except for blanks or control characters), and date must be entered as `MM/YR` or `MM/DD/YR`. If no valid date is found, the date the document is loaded is used.
- Next comes text, which may contain bravo paragraph information (which is ignored). However, documents must contain less than 4,000 bytes. Additional characters are ignored and a warning message is output.
- Documents must end with the symbol `#` followed immediately by `CR` (not control `CR`).

The `DocLoad` program might Swat if data does not conform to the above requirements. In this case, the integrity of the four `FIRST` files cannot be guaranteed. Therefore, the `FIRST` files should be backed up before updating. Any messages output to the display during `DocLoad` are saved in the file `DocLoadLog.first`.

The time needed to load documents depends on the length of the document, but a typical 200 word abstract might require about 1.5 minutes. In the beginning when the B-tree is very small, documents will load much faster.

Currently, data bases must reside on the operating system disk, running on either a Diablo model 31 or model 44 disk drive. Only one `FIRST` data base is allowed per disk. The total space needed for a `FIRST` data base is approximately twice as large as the raw input text.

Searching a FIRST Data base

The program `First` is used to search a FIRST data base. All user inputs must be terminated by escape. FIRST prompts the user for a query and a date or date range. If a single date is entered (MM/YR), only documents from that date forward will be retrieved. If two dates are entered (separated by one or more spaces), only documents from within that range of dates (inclusively) will be retrieved.

The query is then processed by the stemming algorithm exactly as documents were processed when added to the data base. Any query words not found in the B-tree dictionary are indicated. Neither these words nor any suffix variations of them occur in any documents, so they do not contribute to the retrieval of any documents. If the user is satisfied with the query, "y" is typed (followed by escape) in response to the prompt "OK to search?".

The system then indicates the number of documents which satisfy the date specification and have a non-zero similarity score with the query. The output window is used to display the document-query score for all retrieved documents. All scores are between 0 and 1, with a higher score indicating greater similarity. The user can scroll this window (see next paragraph for description of scrolling) to view all the scores in order to determine a cutoff point for displaying documents. After entering escape to stop scrolling, the user is then prompted for the number of documents to be output.

Documents are output using a scrolling package which allows some control over display of documents. The left (red) mouse button can be used to scroll the display forward; e.g., holding down the button scrolls the display forward, and releasing it stops the scrolling. The middle (yellow) mouse button operates like the yellow "bookmark" button in Bravo. The right (blue) button has no effect. When the user is finished looking at retrieved documents, escape allows another query to be entered. However, entering a new query will destroy the output file for the previous query (see below).

Any documents which the user requested to have printed are output on the file `Scratchfile.first`, whether or not they were actually viewed on the screen. Hard copies of this file may be obtained using Bravo or any other program accepting text files. However, this file is deleted the next time `First` is executed.

A FIRST data base consisting of abstracts, keywords, and bibliographic information from the *Communications of the ACM* (1970-1977) is available for searching using `First`. In addition to `First.run`, the following files must be retrieved from Ivy station WRC:

```
<dattola>DocAbs.first -- 1902 pages
<dattola>DocMat.first -- 622 pages
<dattola>BFAtree.first -- 692 pages
<dattola>Auxtree.first -- 32 pages
                Total -- 3248 pages
```

Since some of the files are so large, users outside of Webster should retrieve the files during off-peak hours.

WHOLE ALTO WORLD NEWSLETTER SUBSCRIPTION RENEWAL FORM

NAME _____ MAIL STOP _____

ORGANIZATION _____

What do you use the Alto for?

What sections of the Newsletter do you usually read?

What sections of the Newsletter do you seldom read?

What additional topics would you like to have included?

RETURN TO: Frank Ludolph, PARC (Internal mail)
or c/o Xerox Research Center (U.S. Mail)
PaloAlto, Ca. 94304
or Message: < Ludolph >

Whole ALTO World Newsletter

Technology and Tools

XEROX

July 31, 1978

SPECIAL ANNOUNCEMENTS

NEW WHOLE ALTO WORLD CHAIRMAN NOMINATED - Jim Iverson of the Webster Research Center has been nominated to succeed Liz Bond as chairman of the Whole Alto World. A special committee, appointed at the last WAW meeting for this purpose, met in late June to discuss potential candidates. They selected Jim based on his active participation, energy, and expressed desire to continue and expand the activities of the Whole Alto World. The committee's recommendation will be presented for ratification at the next WAW meeting in October.

GENERAL NOTES

ALTO NETWORK EXPANDS - Xerox Computer Services in Los Angeles and the Office Systems Division in Dallas are joining the Alto network. XCS became an operational member this month. They are connected through the ASD/EI Segundo Gateway (a map of the current network is attached to the Newsletter). XCS has two machines at this time and will be doing some MESA programming in conjunction with SDD.

OSD will be come up this month. They will operate over part of the bandwidth of an already existing line to WRC. Dallas currently has three machines, two of which are used by OSD for human factors work and one by DSD for design.

SUBSCRIPTION RENEWAL RESPONSE - Thank you all for the strong response. Some good suggestions were received as to how the Newsletter might be expanded. They will be reviewed during the next month along with a discussion in the August edition of the more popular requests. An immediate result is the addition of a new section, MARKET PLACE. Several people indicated on their subscription renewal that they would like some sort of "want ads" for software and/or hardware. So beginning with this issue, MARKET PLACE becomes a regular section.

MARKET PLACE

Market Place provides a forum for Alto users to make offerings and requests for Alto related hardware and software. To place an "ad", send the text to the coordinator, Frank Ludolph (PARC), message <Ludolph>, or phone Intelnet 8*823-4356.

ALTO MAGNETIC TAPE CONTROLLER - ASD has developed a 1600 bpi mag tape controller from a David Boggs' design. The prototype is now at XEOS and work is in progress on a software handler. The build has already begun. If you would like to participate, contact Jerry Palbilki at Intelnet 8*823-1637. Cost is \$2200 for parts and labor plus a share of the non-recurring engineering cost (\$20K to be split among all buyers). So far there are 9 buyers. Act now.

NON-GLARE SCREEN - ASD has a need for Alto displays with non-glare screens to fill a customer's request. These monitors were supplied with 4th build Altos. They can be recognized by the diffused appearance of images reflected off the screen. ASD will trade displays from the 8th build one-for-one. The 8th build displays have a user-accessable brightness control. The exchange is for the display only, not the entire workstation. Contact Chuck Anthony at Intelnet 8*823-1956 or message <Anthony>.

TOOLS

HARDWARE

DOVER II AND SPRUCE - The Dovers built by SPG have some minor engineering changes that slightly alters their interface characteristics. Old versions of SPRUCE will not drive Dover IIs properly (the first page images on the drum but paper does not feed). Newer versions of SPRUCE (7 and 8) will operate properly with both old and new Dovers. Coupled with the recent font and Press changes, it is recommended that all sites use version 8 of SPRUCE (check the output break page for the version number).

MAINTENANCE NOTES

BEWARE OF RF SOURCES - No one loves an arc welder. One such machine has disrupted digital and video activities at the PARC facilities inspite of a couple of moves. Electrical arcing can produce a large amount of energy throughout the radio frequency spectrum. Alto related symptoms are disk read/write errors on Diablo and Trident drives. Model 31s shielded within the Alto case have not experienced problems. However, external drives, as in dual-drive systems, must be grounded to the Alto chassis in the presence of heavy RF otherwise write operations may return 'not-ready' yet still write garbage. The Trident doesn't fare as well. Its symptoms, random checksum errors on reads, can be reduced but not eliminated by using an earth ground. In neither drive does the metal case provide adequate sheilding.

To test for excessive RF energy, lay a long (15 foot) wire across the floor and attach it to a scope input. Interference problems have been encountered starting at about the 1 volt level. Frequency sensitivity has not been measured.

Other RF sources have been recorded including PUP Ethernet transmissions (.2 volts), the display's fly-back transformer (a 40 μ sec pulse), and display switching through a multiplexer (1 volt). The display related pulses have caused disk errors (watch the screen of an Alto running TRIEX on a Trident for error messages).

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local IVY server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [XEOS].

PUP PROTOCOL DOCUMENTATION - In keeping with the spread of Spruce printers and the demise of Ears, Ed Taft recently converted PUP protocol documents from Pub format files to Bravo format and regenerated all Press files using the new fonts. Most importantly, the documents themselves were updated. They are available from [MAXC]<Pup>.

IFS or IVY - Some time ago it was decided to rename the IFS file server software to IVY. However, since the name IFS is so pervasive in the documentation and folklore, IVY never caught on and its use only caused confusion. Whole Alto World documents will, from now on, use 'IFS' to mean the system software and 'IVY' to indicate the file server at PARC.

NEW RELEASE: ERL - Tom Moran of PARC and George Robertson of Carnegie-Mellon have produced a new general programming system on the Alto based on Alto L*. It was designed for

running small user interface and user performance experiments which require real-time control with no unpredictable system events (e.g. page-swapping or garbage collection). ERL is a list-structured language with homogeneous program and data representations with easy-to-use programming facilities for graphics and interrupts. The ERL system runs entirely within memory (it is interpretive and moderately fast), and offers facilities for debugging, editing, and filing programs. Documentation can be retrieved from [MAXC]Lstar>ERL.press.

ReReleases - Subsystems

ANALYSE - This release fixes some subtle bugs. Retrieve [Maxc]KSIL>ANALYSE.run. The documentation is unchanged

BUILDBOOT - This release contains internal changes and cleanup. The documentation is unchanged.

CONDENSE - See the MENU entry under ReReleases - Packages.

DRAW - Version 4.3, dated June 23rd, now provides Spruce/Press compatibility and supports the recent Spruce printing changes. Retrieve DRAW.run or load DRAW.dm if retrieving for the first time.

FIRST CACM DATABASE - A new version of the CACM database (1970-1977) has been loaded that enables psuedo-bibliographic searches. (Extracts containing the requested author's name will also be reported.) Retrieve [WRC]K<Dattola>docabs.first, docmat.first, bfatree.first, and auxree.first. The FIRST system and documentation can be loaded from [WRC]K<Dattola>FIRST.dm

IFS - The IFS FTP server now generates a message in response to a "rename" command if the "to" file already exists. This will be effective when IFS is updated by the local administrator. The documentation is being revised.

IFSSCAVENGER - A bug important in scavenging non-IFS Trident disks has been fixed. Retrieve [MAXC]IFS>IFSSCAVENGER.run.

MENUEEDIT - See the MENU entry under ReReleases - Packages.

PEEKUP - This version has been updated to recognize newer PUP types. The documentation is unchanged.

PREPRESS - This release fixes minor bugs. Retrieve PREPRESS.run. The documentation is unchanged.

PUPTEST - This subsystem is undergoing internal cleanup and is being enhanced with additional Alto network drivers. The documentation is unchanged.

ReReleases - Packages

INTERRUPT - Calls to DisableInterrupts now returns TRUE if interrupts were set, otherwise FALSE. Current users of the package are unaffected. The documentation, Interrupts.tty, is undergoing revision.

MENU - Version 1.4 contains several changes including standard file names, inactive boxes, group box moves, and a box outline specification. The new version is not compatible with previous menu definitions, however, a /U switch for MENUEEDIT can be used to convert. Load <Alto>MENU.dm

Whole ALTO World Newsletter

and retrieve MENUEDIT and CONVERT. A memo outlining the changes can be retrieved from <AltoDocs>Menu-News.bravo. Revised documentation is on Menu.bravo.

TECHNOLOGY

It is well known to Alto users that Xerox is interested in developing Office Information Systems. Important in the success of this effort is a broad awareness of the behavioral effects the installation of an OIS is likely to entail. The first paper, "Behavioral Implications of Office Information Systems", describes the different ways OIS can impact client organizations. The second paper, "Some Considerations for Office Technology", reports on observations made in a working office from three different perspectives and analyses the potential effects of OIS on the office's operation.

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC]<AltoDocs>WawNewsM-YY.press or may be obtained from the editor, Frank Ludolph, XEOS, by messaging <Ludolph> or calling Intelnet 8*923-4356.

Behavioral Implications of Office Information Systems

A Report of a Study Conducted by the Office Research Group

May 1978

Abstract

OIS includes a range of applications from routine data-handling to highly sophisticated control of information, and OIS will inevitably impact client organizations along many dimensions. The main conclusions of this paper are: the composition of the client firm's work force will change; the OIS hardware may produce unforeseen behavioral effects; extremely close monitoring of employee performance will be possible; managers will have data on how they are using and processing their data; and managers will reject an OIS if it requires too much change on their part.

We feel that if Xerox is to pursue the development of OIS, it has no choice but to develop simultaneously a strong capability to deliver training and consultation to purchasers of OIS. We also feel the implementation problems of OIS cannot be adequately treated through a standardized "blueprint" which is applied to each customer.

If we are to counsel users these issues must be squarely addressed. We hope this paper will focus attention, stimulate a dialog, and precipitate action.

XEROX

PALO ALTO RESEARCH CENTER

3333 Coyote Hill Road / Palo Alto / California 94304

This paper points out some of the major ways in which the Office Information Systems (OIS) now under development will both impact upon and be impacted by behavioral factors. Behavioral factors include elements of employee motivation, job satisfaction, hiring, training, supervision, and organizational design which will interact with OIS. The primary focus of this summary is on the ways in which the implementation of an OIS will interact with behavioral factors in a client organization. However, the summary will also mention some implications for the role of the Xerox team which installs such a system.

This report is divided into three major sections. The first section briefly describes the broad characteristics of the Office Information Systems currently under development. The second section describes the behavioral implications of these developments. The third section presents two key considerations which have emerged from these discussions.

This paper summarizes a series of discussions held by members of the Office Research Group (ORG) of the Palo Alto Research Center, Xerox Corporation. This study included a review of the Xerox experience with word processing centers (WPC) and with administrative processing centers (APC). It also included some consideration of the relevant behavioral research, with guidance from an outside consultant, Professor William G. Ouchi of the Stanford University Graduate School of Business, and a research associate from Stanford University, Melanie Powers. These inputs are reflected in the analysis and conclusions reported here.

The questions raised in this paper represent the best guesses of the Office Research Group concerning these behavioral issues. At this point, some general characteristics of the OIS being designed have clearly emerged, although their final form may be subject to change. The exact hardware and software which will ultimately reach the customer is yet undeveloped except in prototype, so that the conclusions reached here are based on broad outlines rather than on specific properties of OIS.

I. Characteristics of Office Information Systems

OIS includes a range of applications from routine data-handling to highly sophisticated control of information. The routine data-handling properties of the system include the entry, storage, manipulation, and retrieval of data of the type which is currently done by clerical employees. An OIS can carry out these routine functions more efficiently than existing methods of doing this work.

However, the distinctive characteristic of OIS is its capability for information control. Systems currently under development have the ability to capture a previously infeasible spectrum of functions governing the location, use, and flow of a great range of information. For example, a sales representative will be able to know in an instant the status of a customer order. Supervisors will be able to monitor precisely the volume of work, type of

work, and error rate of clerical employees who enter data into an accounting system. Personnel departments will be able to design and immediately implement reporting forms required by new legislation.

The various OIS packages currently under development will combine data management capabilities with the additional capacity for information control. It is an inherent characteristic of these systems that while they store and process records, they will simultaneously be capable of tracking the speed and efficiency of the system operators.

II. Behavioral Impacts of OIS on Client Organizations

OIS will inevitably impact client organizations along many behavioral dimensions. Our objective here is to anticipate the general nature of these impacts and to suggest the outlines of a general strategy for coping with them. The most important issue is our firm belief that these behavioral issues will impact each client in a unique manner. OIS is being designed in such a way that a client will have a great deal of flexibility in adapting it to his existing management practices. Despite this flexibility, however, OIS will clearly require organizational and managerial changes of some sort in each client organization. We do not believe that a standardized approach to these unique customer needs will be adequate. No one approach, such as standardized APC's or job enrichment, will fit all customers. Therefore, Xerox will probably have to be prepared to support each OIS installation with a mixture of training, consultation, and other custom-designed help.

Even the simplest OIS installation will have some behavioral impact on a client firm. Some broad generalizations are possible.

The composition of the client firm's work force will change.

Pilot projects suggest that we may be close to achieving our objective of developing a system so simple to use that a typical secretary or clerk can become competent with a few hours of training. In this event, most installations will have no need to hire specialized OIS operators.

In some cases, however, users will want to acquire OIS specialists, possibly creating problems of job grade, of compensation, and of reporting responsibility. The OIS specialist may, in some cases, outrank the OIS supervisor, especially when that supervisor was formerly head of the clerical department which adopted OIS. If these relationships are not handled properly, they are sufficient to cause the installation to fail, as was apparently the case in some WPC's.

In other cases, the user may choose to emphasize only the simplest, most routine uses of OIS. In such cases, if existing clerical or secretarial personnel are transferred to these jobs,

they are likely to feel that they have been downgraded to less interesting work. If less skilled employees are hired to operate OIS, then the former secretarial and clerical employees will be subject to termination. Either of these contingencies, improperly handled, can cause the system to be rejected.

Work force changes brought on by OIS can cause a user more trouble than it is worth. Clerical employees who fear the impact of OIS on their jobs can sabotage it. Effective Xerox support may be able to forestall these behavioral problems.

The OIS hardware may produce unforeseen behavioral effects.

In some cases, OIS equipment may be noisy, CRT's may produce eye fatigue for those who use them constantly, or other physical arrangements of the system may reduce the opportunity for social interaction between employees. Particularly among those workers whose jobs are intrinsically boring, any increase in individual isolation or other significant change in the social aspects of work may make the job unbearable. OIS should not be allowed to become a space-age "sweatshop".

The potential impact of these issues should not be underestimated. Many clients have noted that the addition of a Xerox copier alters patterns of social interaction in an office typically increasing social contact. If a Xerox OIS were to similarly *decrease* the frequency of social contact available to secretaries or to clerks, they would almost certainly rebel in one way or another. For example, if OIS equipment were placed in a manner which made it difficult or impossible for operators to exchange small talk, and if the company were one in which clerks have little or no influence over supervisors, then the result might be greatly increased rates of turnover such as in some WPC's.

Extremely close monitoring of employee performance will be possible.

The information-capture capabilities of OIS will make it almost automatic to retain, in extreme detail, quantitative data on the job performance of all personnel who work with the system. It will be possible for managers to intrude into the private work space of an individual to an extent unapproachable with current technology. *The potential behavioral impact of this innovation may be more important than any other.* Supervisors in some cases may react like a small boy who has just discovered a hammer -- he finds that everything needs hammering. This problem will be especially acute in the case of first-line supervisors. This position is almost universally one in which the supervisor is engaged in at least some conflict with subordinates. The subordinates (clerical or data-entry workers) are engaged in dull jobs and thus frequently seek ways to divert themselves, which are commonly perceived by a superior as "goldbricking". A supervisor in this kind of situation, given the ability to monitor completely every aspect of the performance of every individual subordinate, could quickly develop an inappropriately tough, solely output-oriented managerial style that would

result in a very high level of turnover and low employee morale. A skilled Xerox representative can help higher levels of management in the client firm to anticipate issues of privacy and avoid problems of counterproductive over-monitoring.

Managers will have data on how they are using and processing their data.

Lower and middle level managers in client firms will be able to monitor not only the performance of their subordinates but also the performance of their work systems. They will be able to track the rate at which bills are arriving or being paid, the status of the distribution of new customer account data to field offices, or the average age of credit data on customer accounts. Since most lower and middle level managers are not accustomed to data manipulation and use, the nature of their jobs will change. As the experience of many companies in introducing computerized management information systems over the past two decades has shown, we can expect two general effects: (1) most managers will not use the system up to its full capabilities; (2) most managers will change their behavior to some extent and will make use of the system in at least a limited way. Although we can be relatively confident that managers can and will adapt themselves to OIS, we cannot expect anything approaching optimal usage without help. The companies which have been most successful in selling business computers have devoted a great deal of their resources to developing the means of training and consulting with their clients on these issues. Xerox may have to mount a serious effort of this sort.

Managers will reject an OIS if it requires too much change on their part.

Managers, unlike lower-level employees, have the power to reject an office system which they don't like. Therefore, it is critical that the design of those features with which they come in contact take into account a deep and complete understanding of the needs and tastes of managers in potential client firms. (Relying upon introspection is particularly dangerous in this case, since most Xerox managers are far more accustomed to information technology than is the average manager). Perhaps the most innovative characteristic of OIS is its adaptability to the working style of most managers. One of our primary objectives must be to design a system which a manager will want to use, one which fits into his habitual modes of thinking, communicating, and problem-solving. However, the evidence of behavioral science research and the history of business computers and WPC's all indicate that an apparently small change in habits or relationships will drive many managers to reject a system despite its overall advantages.

For example, an OIS which permits greater efficiency in creating and storing travel expense reports but which transfers even a small fraction of the task from the secretary to the manager is not likely to be welcomed by managers. An OIS which gives others ready, convenient access to these same expense reports may also be unwelcome.

Not all behavioral impacts will be negative. It would be a mistake to conclude from the above that we must try to implement OIS without intruding any change in the office sociology. It would be foolish, as well as impossible, to try to preserve, say, the exact pattern of communication paths that support the existing friendship groups. It is inherent in OIS that it will intrude into the communication patterns of an office. Indeed, if OIS does not alter office communication structure, then it accomplishes nothing. From a behavioral point of view, however, a change in patterns of communication is almost always disruptive.

In most offices, the disruption will be temporary, and a new and equally satisfactory pattern of social and business communication will grow up around the OIS. In such cases, the critical task is to support the client in a manner which avoids unnecessary disruption and which passes through the necessary disruption as quickly and as effortlessly as possible.

In some installations, the client may explicitly wish to alter permanently patterns of communication in a major way. Here the Xerox task will be more complex.

In summary, an OIS is a process, not a machine. Each OIS installation will include elements of hardware and software that will make it at least somewhat unique, as is the case in computer systems. Each client will have organizational and managerial conditions that will make it unique. A flexible approach to implementation and a commitment to support the whole OIS -- both technical and behavioral -- are appropriate.

III. Implications for Internal Xerox Organization

The analysis above leads to two key questions that we would like to emphasize.

Should Xerox develop internal training and consulting capabilities?

We feel this is the most critical question. We feel that if Xerox is to pursue the development of OIS, it has no choice but to develop simultaneously a strong capability to deliver training and consultation to purchasers of OIS. An integrated office information system must be supported by technical training, management training, and some specialized, high-quality consultation on organizational and behavioral issues relevant to office information systems. We are not suggesting that Xerox has to become a giant consulting firm. Rather, we are suggesting that we develop a high level of expertise in a relatively narrow, specialized area of training and consultation. Implementation of OIS will be a significant change in at least some features of the managerial process in the client organization. Since most clients will be inexperienced at managing this kind of change, and since Xerox will quickly become familiar with it, it seems reasonable for Xerox to provide these support services.

Should Xerox design a standardized installation format?

The implementation problems of OIS cannot be adequately treated through a standardized

"blueprint" which is applied to each customer. The systems being developed provide sufficient flexibility to be adapted to the needs of a wide variety of users with a wide variety of information needs and managerial styles. Nevertheless, a properly "tailored" OIS, like a properly matched computer system, should provide a client with effective, trouble-free service. However, this flexibility means that an inexperienced client or Xerox representative may be overwhelmed by the variety of forms which the final installation is able to take.

IV. Conclusion

We believe that Xerox will have a superior technology to offer to customers, one that will give them both immediate, tangible cost savings and enhanced managerial capability. We further believe that our proprietary system will be superior to any which our competitors will be able to offer. However, it is only by considering key behavioral issues that we will realize success in the marketplace.

Computer application projects often proceed without regard for the behavioral implications of installation and use. This has been the case for OIS. There is, however, a clear choice. On one hand, OIS could be sold as a collection of hardware. On the other hand, OIS could be sold together with consultation concerning the behavioral and managerial implications of installation.

This choice must be made soon. These issues must be squarely addressed. We hope this paper will focus attention, stimulate a dialog, and precipitate action.

There are opportunities for fruitful interaction among Xerox groups concerned with OIS. A short-run objective would be to exchange information concerning the technical and behavioral characteristics of OIS. A long-run objective would be to understand more effectively the kinds of future OIS developments which are technically feasible and are compatible with larger Xerox objectives.

Some Considerations for Office Technology

BY Cheryl Crawley, Katherine Newman and Allen Sonafrank

October 1977

University of California, Berkeley

Abstract

The observations and analyses presented in this report summarize the authors' findings from a six-week fieldwork period. These observations are intended primarily for the use of those involved in developing new types of office technology. The authors believe that it is vital to understand the working environment into which new forms of technology may be introduced. This project could not have been completed without the invaluable cooperation of their members of the three Customer Service Divisions with whom the authors worked.

XEROX

PALO ALTO RESEARCH CENTER

3333 Coyote Hill Road / Palo Alto / California 94304

Experience has shown that the introduction of new office technology can have important effects on the daily operation of bureaucratic organizations. Technological innovations may require the reorganization of work groups, reallocation of responsibilities, and major changes in the nature of the tasks that employees currently perform. The effects of these changes can be minimized, however, by anticipating the impact of new technologies and by designing systems that are sensitive to the environment into which they will be introduced. Therefore, in order to create office systems that will be successfully accepted, researchers need to understand the working environment and the people who are part of it.

This project was initiated by the Office Research Group of the Palo Alto Research Center with a particular goal in mind: to familiarize individuals involved in creating new technologies with the work situations and the office people at the company being studied. To this end, anthropologists from the University of California at Berkeley joined three different company branch offices for six weeks of field observation. They participated in the daily activities of the customer service division of their respective branches in an effort to learn as much as possible about the operation of this one department. The researchers were quickly incorporated into the daily routines of the customer service divisions and were thus able to observe ongoing events unobtrusively. This report is a summary of their combined research findings.

The term "working environment," is open to several interpretations. For example, the "ecology" of the work place could be studied to determine how physical location and population density affect the behavior of employees; alternately the psychological aspects--the effects of personality traits on job performance and motivation--could be examined.

Each of the three researchers on this project studied the working environment from a different perspective. Like the approaches mentioned above, each of these highlighted certain aspects of the field situation. By using three complementary "definitions" of the work situation, a more complete and holistic description of the field setting was produced than would have been possible using only one perspective.

The first of these approaches is drawn from an area of social science known as "formal organizations theory." From this viewpoint, the work environment consists mainly of a series of incentives, controls, and constraints on the behavior of individuals in the office. These controls are generally instituted on a formal basis; that is, they are developed by management in order to encourage individuals to cooperate in the pursuit of organizational goals. However, because these formal controls elicit *informal* responses from employees, the formal organizations perspective emphasizes both the formal structure created by behavioral controls in the work place and the typical response patterns that develop in reaction to those controls.

The second approach focuses on socialization in the work place. Here the researcher's goal is to describe how individuals who enter the customer service division become established members of the organization. What kind of "world view" do people adopt as

they become absorbed into the ongoing activities of the branch? What sorts of social rules must they master in order to function effectively as employees? In short, how does one go about taking on new roles in the branch? Whereas the first approach deals with the structure of the organization, the second deals primarily with the way in which participants learn and interpret that structure.

The daily tasks that individuals perform should be understood within both of these contexts. Thus the third research perspective, which examines the actual activities of customer service division employees, particularly the customer assistants (billers), describes the decision-making processes that are central to the work. Drawing on an important area of social science known as "decision theory," this approach attempts to identify the significant choices that individuals must make in the course of fulfilling their responsibilities. In particular, the emphasis is on the information used and the factors that influence decision making on the job.

These three perspectives provide a detailed description of the working environment of the customer service division in each of the three branches. Beginning with the most abstract perspective and moving toward the most concrete, a comprehensive picture will be provided of the kind of situation into which new forms of office technology might be introduced.

1. Control Strategies and Response Patterns in the Customer Service Division

Every bureaucracy must deal with the problem of gaining the cooperation of its members; that is, the interests of each individual must somehow be channeled so that the goals of the organization are reached. There are a variety of strategies that business organizations use to gain the compliance of employees. For the purposes of this project, two particular control strategies should be mentioned--utilitarian and normative. Utilitarian organizations are those that use the incentive of financial reward in order to obtain desired employee behavior. People tend to cooperate because they know that they will receive pay raises, bonuses, or some other form of monetary reward. In contrast, normative organizations rely on the fact that employees feel a sense of personal identification with and loyalty to the organization. Individuals in normative bureaucracies work hard because they believe in the goals of the organization and see themselves contributing significantly to its activities.

Each of these general control strategies has advantages and disadvantages. Utilitarian control works best in situations where a firm organizational hierarchy exists; individuals work to fulfill production quotas and are content with financial reward instead of influence in the decision-making process. Normative control works best when individuals are more autonomous and can be involved in decision making. People who identify personally with the goals of the organization must be made to feel that their opinions count in important decisions. Normative control can only be effective when the hierarchy of the organization

is willing to relinquish some control to those at lower levels.

A mixture of the two control strategies was evident within the customer service division of each branch. Individuals were financially rewarded for performing at a high level. At the same time, they were strongly encouraged to consider their jobs as careers, instead of as ways just to earn a living. This combination of strategies was implemented by the use of targets, which measured absolute production (e.g., dollar amounts billed, accounts receivable collected) and efficiency (amount of time required to transfer information from documents into the office information system), and by the use of employee evaluations, which encouraged particular kinds of behavior.

The target system is a formally sanctioned method of measuring employee productivity and efficiency. However, the particular kinds of targets used in the customer service division (and elsewhere in the branches) have certain very important informal consequences. In order to understand the nature of these informal responses to targets, one significant aspect of this measurement system should be noted: Every position in the customer service division carries certain responsibilities that cannot be met without cooperation of individuals over whom little direct control can be exercised. Thus, for example, equipment order entry (EOE) personnel are responsible for processing installation and cancellation of orders and are targeted on the time that elapses between the date of a request for action and the date on which the computer system recognizes that such a request has been made. However, in order to fulfill this efficiency criterion, the EOE clerk must have the cooperation of the salesperson with whom an order has been placed. Order documents must be turned in on time and must include all the necessary information before they can be dealt with properly. The salesperson is responsible for completing the documents correctly. Should he/she fail to do so, the order is in danger of being processed late, and the EOE clerk will be held responsible. The individuals in equipment order entry have little control over the behavior of sales staff and yet they depend on them in order to complete their own responsibilities. Fortunately, the dependency is mutual. The salesperson does not receive credit for placing an order until the computer system acknowledges that order.

What this example indicates is the fact that the target system sets up certain mutual dependencies among members of the branch. Informal relationships arise through which individuals can obtain some leverage over each other and can thus gain the necessary support. These relationships tend to be very strong where the dependencies are mutual, as in the case mentioned above, and somewhat weaker in cases where the dependency is greater on one side than on the other. Given the present targeting system, these informal relationships are indispensable to the smooth operation of the customer service division.

In a purely utilitarian organization, it might be expected that individuals would regard targets somewhat antagonistically since they would represent only the way in which measurement for the purpose of financial reward takes place. Of course some of this feeling exists in the customer service division, indicating the extent to which utilitarian controls are felt. However, the normative side of the branch mitigates against such a

negative attitude. Employee evaluations stress not only target performance, but subjective qualities such as initiative, reliability, inquisitiveness, and creativity. In short, evaluations encourage an individual to invest his/her sense of self-worth in what might otherwise be seen as a routine clerical job. How do these subjective aspects of employee evaluation affect responses to the target system? Two particular response patterns should be mentioned. First, people take on targets as personal challenges rather than as mere measurements of their efficiency or production; they see meeting or surpassing a target as an individual achievement of which they can be proud. Second, individuals regard their jobs and the targets as interesting to the extent that they require problem-solving abilities. In order to meet the targets successfully, customer service personnel must draw upon their knowledge of a fairly complex organization. They see difficult cases (e.g. billing problems, collection problems) as opportunities to exercise their detective skills or as chances to use persuasive abilities in dealing with delinquent customers. In short, the evaluation system rewards people not only for meeting numerical targets, but for displaying high levels of initiative in problem solving.

We mentioned before that normative control works best in situations where a certain amount of authority and autonomy can be granted to lower level employees. Given the pervasiveness of the targeting system and the frequency of employee evaluation, it may be difficult to see how individuals can feel that they have much personal control. In order to understand what supports this idea, the *measurement period must be examined*. The target system is set up to calculate output and efficiency on a monthly basis. Although employees must respond to the requirements of the measurement system, they have considerable flexibility in determining exactly how they will do so, and the length of the target period allows them to formulate individual and group strategies for achieving target levels. To the degree that they are able to decide autonomously how to attack a month's work, they feel that they are in control. The best customer service employee is the one who is able to determine how to meet and surpass targets independent of management control. Employees use this autonomy as a measure of how much their *expertise* is valued. Increased management control over decision making in areas like work pacing and working techniques are seen as an encroachment on employee autonomy and an implicit criticism of the employee's professional judgment. The lesson to learn from this is that high employee motivation, which is the aim of normative control, is obtained by giving employees sufficient autonomy, thereby asserting the organization's faith in their abilities.

Although every individual has some leeway in deciding how to approach monthly targets, it should be noted that variations in the structure of work groups can influence these decisions. There were two basic types of organizations evident in the work groups (or functions) in the customer service divisions. Groups could be organized according to a functional division of labor in which each individual performed only part of a sort of assembly line process. Alternately groups could be set up in a pooled production system with each member performing exactly the same duties but on a different set of materials. This was usually the case in each of the three customer service divisions. Each employee

handled one or more sales teams and did everything necessary (within the bounds of his/her function) for those teams. For example, each collection representative handled account collections for the sales teams that were assigned to him/her. In contrast, several work groups were organized along the lines of a functional division of labor. Thus, for example, some work groups divided tasks so that processing any one document required the coordination of two or three individuals in a sequential order.

Certain important differences tended to arise from these two types of group organization. Pooled production tends to lead to the formation of informal hierarchies *within* the function or work group. The most experienced and/or successful individuals were likely to assume the role of team leader, thus taking on more responsibility for communicating with management and devising work strategies. Since pooled production does not encourage specialization and requires all individuals in a work group to be able to perform *all* necessary tasks, those who were especially good at it were able to rise to an informal but recognized position of leadership. Furthermore, the broad capacities of people in pooled production units enabled work sharing to take place, particularly under high stress conditions created by absences, heavy work loads, and so on.

Groups with a functional division of labor tended to encourage specialization among the employees, who were experts in their own particular area of work and were much less familiar with the activities of their co-workers. Thus efficiency could be very high as long as each group member was taking care of his/her part of the overall work process. As a result, however, work sharing could only be practiced on a limited scale. Furthermore, informal hierarchies were less likely to develop because it was difficult for one individual to gain the combined expertise of his/her specialized co-workers. In short, the functional division of labor was efficient under optimal conditions, but tended to be less so under stress.

So far certain aspects of the formal organization of the customer service division have been considered, but the question remains regarding the relevance of this information to those in the office technology field.

The following is a condensation of some points drawn from the field research that may be important in the introduction of new forms of office equipment:

- 1) Normative control is most appropriate for organizations whose tasks require a high degree of initiative; however, in order to implement normative control successfully, employees must feel that they have some autonomy and authority, and that they are valued by the organization.
 - a) Office systems that limit the frequency with which productivity and efficiency are measured will, by deemphasizing close surveillance, encourage people to feel autonomous and more highly motivated. The longer the interval between measurements, the more opportunities individuals have for self-pacing. Monthly

evaluations will be responded to more favorably than weekly targets; and daily targeting might greatly reduce employee motivation.

- b) Individuals feel they have authority when they have a good deal of flexibility and control over their work strategies. Thus, office systems that increase this flexibility are likely to be received positively.
- 2) Jobs that require high initiative for some purposes, but that also entail considerable routine work, require individuals who are committed to their jobs and to the organization itself. When comparatively routine jobs are filled by well-educated individuals who need intellectual stimulation, the employees will be most interested in the aspects of their jobs that involve special skills.
 - a) Problem solving in uncertain situations is valued by such individuals.
 - b) Office systems that preserve areas in which problem solving is necessary will be viewed as adding to the intellectual interest of routine jobs.
 - 3) Complex organizations that use targeting systems encourage the formation of informal working relationships by creating interdependencies in and among departments.
 - a) If an automated office system alters the flow of information within an organization, changes can be expected in the nature of these informal relationships.
 - b) It seems reasonable to expect that wherever new dependencies for information and influence are created, new informal relationships will be formed. Thus short-term disruptions in the flow of communication and the strength of social networks may settle down as the necessary social adjustments to changes in dependency patterns are made and new informal networks are built up.
 - c) Wherever possible, new technologies should strive to create mutual dependencies. One-sided dependencies tend to create weak informal relationships and less efficient systems of work management, whereas balanced mutual dependencies encourage strong informal relationships and smooth cooperation between divisions.
 - 4) The division of labor within a work group influences the kinds of informal hierarchies that develop within it.
 - a) Office technologies that lead to changes in the division of labor within work groups will probably cause major shifts in the informal organization of those groups.
 - b) Technologies that replace pooled production groups with increasingly specialized divisions of labor are likely to cause a corresponding decrease in informal work-sharing networks, informal leadership, and flexibility under stress.
 - c) The reverse can be expected of technologies that move in the opposite direction.
 - 5) In organizations that operate under very stressful conditions (rapid employee turnover, heavy work loads, etc.), any change in the environment can be disruptive at first. However, this disruption may not be related to the new technology itself, but to the

pressures of the existing work situation.

- a) These effects might be minimized by adding extra trained personnel during the introduction and adjustment period *or* by reducing production targets for a short time.
- b) Whenever possible, it would be helpful if new technologies were introduced first into stable work groups that have had the same members for a long time and in which informal relationships are thus long-standing.

The foregoing recommendations describe some of the ways in which observations from the field settings and analytical techniques derived from formal organizations theory can be combined to generate insights useful in developing office information systems. This approach to the working environment of the customer service division has focused on the effects of various control strategies and the ways in which changes in these strategies might effect the work situation. However, as mentioned before, the working environment can also be interpreted in ways that stress factors other than control patterns and responses to them. The next section of this summary will present some aspects of socialization in the customer service division and some ways in which the structure is internalized by employees as a set of social rules.

2. Office Socialization: Acquiring a World-View in the Customer Service Division

When an individual takes on a new job, he/she (abbreviated in this section to he) is concerned with a great deal more than learning to handle new tasks. The newcomer must also learn the social behavior acceptable in the work environment and must obtain the cooperation of others in a way that will be positively interpreted. Social rules of this sort are part of everyone's general background, but the specific form of those social conventions may be particular to specific situations. Thus the person who steps into an unfamiliar position has to become aware of the social rules that members of a group use to interpret each other's behavior.

Several phases of the office socialization process were distinguishable in the field setting:

- 1) *recruitment* -- individuals are selected who show certain desirable personality traits and/or work habits;
- 2) *introduction* -- the individual begins to learn the structure of the organization and the division of responsibilities, and begins simultaneously to build social loyalties to co-workers and to the division;
- 3) *social adoption* -- the newcomer begins to be accepted by the members of the organization and is included in informal social events; and

- 4) *world-view and value formation* -- the employee, after he is no longer considered a newcomer, begins to see the organization through the eyes of a full-fledged member of the group and thus acquires a recognized and validated self-image and role within the work group.

This four-phase process is experienced by any new participant in the work situation and certain aspects of it may reapply to members of the organization who change roles; for example, an individual who moves from the position of a worker to that of first-line management must be resocialized. He/she must learn the social rules that govern the kind of interaction he engages in with subordinates, or the way in which friendships with higher-ups should be formed and a new peer group found. Promotion into managerial ranks must be reflected socially: he must be able to project a new image of authority and prestige as well as administrative competence in order to be *recognized* as having the social right to claim a new identity within the organization. New managers, particularly those who have been promoted from within the company, must learn how to evaluate individuals who might once have been peers. They need to negotiate for certain scarce resources (overtime, supplies, etc.) on behalf of the group of which they were once a member.

How are these sorts of skills transmitted? For lower-level employees, the peer group provides most of the necessary input, although management may play a role. Information regarding formal responsibilities is transmitted directly by co-workers; established employees will generally guide newcomers in their tasks to the extent that time is available for peer training.

Information regarding social rules tends to be more subtle. Through passing comments made about other divisions, the newcomer will learn about shared attitudes of co-workers toward other parts of the branch. By noticing who goes to lunch with whom he can assess the degree of formality or loyalty that characterize relationships in the office. The more an individual is accepted as a trusted insider, the more likely it is that topics such as loyalty and competence will be discussed openly in front of him.

Social conventions are generally transmitted in these informal and relatively subtle ways. In some of the branches, however, certain aspects of the socialization process have been formalized, or institutionally recognized. Thus, for example, at least one branch has adopted a tradition of a yearly party, the highlight of which is the presentation of mock awards; although this is done in good spirits, the roasting that individuals undergo for various aspects of their conduct emphasizes the social rules that branch members feel strongly about. Pressure can be brought to bear to force behavioral conformity on individuals who have overstepped the bounds of conduct.

The end result of this socialization process is a social identity that is accepted by co-workers and that creates a clear role for the newcomer in the organization. This is a sensitive subject; people are very concerned about their sense of self-worth, their prestige, and their status as insiders. They are sensitive to subtle cues: the degree to which they

are noticed by those who may consider them for promotion, the frequency with which they are consulted for their opinions, and the success with which they achieve personal goals for advancement. This sense of self cannot be sustained unless it is continuously validated by co-workers and by management. Perhaps most important, employees must feel that others recognize their positive potential and respect them for it. This is so important to employees in the customer service division that they constantly monitored formal and informal kinds of feedback for evidence of their self-worth and their social position in the organization.

In order to discover how this sense of self might be affected by the introduction of new forms of office technology, the fact that most of the socialization process is carried on through frequent face-to-face communication should be considered. The importance of this was certainly clear to the researchers, and the fact that it is equally clear to employees is demonstrated by their preference to discuss things with their supervisors in person instead of by formal written means. They viewed written communication as a last resort. Supervisors who required formal memos in place of informal contact were thought to be keeping people at arm's length. Two points should be made about this preference for face-to-face communication. First, a good deal more than business information is transmitted on these more informal occasions; individuals are given a sense of their status in the organization, the temperament of their co-workers on a given day, and some reinforcement for their conduct. Second, all types of communication (notices on walls, memos, chats, conferences) are endowed with social significance by the employees. When a supervisor begins to use a memo system instead of less formal modes of communication, the employees consider it an indication of a change in the nature of work relationships. Subtle nuances are discerned in the *form* as well as the *content* of communication.

Perhaps the most important point to note is that, as in most normative organizations, the individuals in the three field sites were very sensitive to the way in which their self-worth was validated by co-workers and superiors. For example, someone who feels himself to be a valuable asset to his work group must receive corroborating evidence from those around him. He must be made to feel that his self-image is shared by others. The process of office socialization might be regarded as a way in which one builds an identity in the work place, and this identity must be reflected in the feedback an individual receives from other people.

Introducing new office systems may change the frequency and kind of communication used in situations such as those described in this report. Because the socialization process is so sensitive to communication modes, it could undergo significant alterations as well. If, for example, face-to-face contact decreases in favor of machine-mediated contact, employees might be disconcerted by the lack of the kind of feedback mentioned above. Conversely, if the new forms of technology still require face-to-face contact, the socialization process will probably be less subject to change. In any case, it should be kept in mind that *any* change in communication patterns is interpreted by employees as reflection on their corporate identity.

The "corporate person" must thus be concerned with a multitude of problems, from working in the context of structural dependencies to constructing an identity through office socialization. Last but not least the individual must deal with the actual tasks to which he is assigned. As will be seen in the next section, the management of daily tasks is a complex process that involves many intricate decisions.

3. Decision Making in the Work Place

The nature of the clerical work in customer service division is described in a wide array of corporate manuals and memos that detail policy and procedure for both new and experienced organization members. No matter how detailed, however, these materials express what *should* be done, what actions *should* be taken, and what procedures should ideally be followed; they lay down guidelines for workers so that each individual can make a rational contribution toward fulfilling organizational goals.

This section of the summary, by way of contrast, describes what *actually* goes on within one particular work group. The level of detail required to describe decision-making processes made it necessary to focus on a single activity within the customer service division. What follows is therefore concerned primarily with decisions involved in adjusting accounts in response to customer inquiries; this is one of the chief responsibilities of Customer Assistants (billers). These decisions are routine in the sense that they occur frequently and require little managerial attention. They are cumulatively very important to the corporation because of their direct impact on customer relations. Furthermore, as will be demonstrated, the process of making these decisions involves the individual in ongoing negotiations in which interpersonal factors such as reciprocity, conflict, and cooperation emerge as primary considerations that influence decision outcomes.

These specific results can be generalized by a demonstration of the manner in which organizational structure and goals emerge as considerations in clerical decision making, and the manner in which they affect personal definition and advancement, conflicts over division of labor, and allocation of time. Inputs to clerical decisions can come from several sources, and the incoming information must be evaluated by the clerical worker in terms of the inferred goals of the person supplying the information. Her inferences are based on the position the informant holds in the organizational structure and on the history of his or her relationship with the decision maker, as well as on factors more narrowly confined to the immediate situation. Projected decision outcomes are weighed, not only in terms of resolving the immediate technical problem, but also in terms of self-advancement, self-protection, reduction of future work load, creating good will and, especially, educating people in other positions about organizational problem areas, opportunities, limitations and mutual responsibilities.

In order to see how these considerations are involved in the process of resolving customer

inquiries, the activities that are required to complete such a task must be examined. The billers' main task, evaluated by means of the unbilled revenue target, is to solve problems listed on a series of computer reports that become available at predictable intervals throughout the month. In addition, billers must find time to fulfill their secondary responsibilities, of which the most important is resolving customer inquiries that reach them through several formal and informal channels (media): telephone, letters, memos, notes, documents such as returned invoices, and face-to-face communication. Two essential elements in defining an inquiry are: (1) a customer's belief that a problem exists; and (2) the fact that the customer has asked a company representative to explain or solve the problem. In practice, inquiries that reach the billers usually concern a customer's account with the company and require explanations of billing problems or adjustments to the account. This discussion focuses on cases that require, or seem at first to require, adjustments to the customer's account. The following decisions are analyzed:

1. What initial response to make upon receiving the inquiry.
2. When to work on resolving inquiries.
3. Which cases to resolve first.
4. Where to file the completed adjustment package.

Decision 1: What initial response to make upon receiving the inquiry.

This is a key decision and it shapes the process of accepting and recording inquiries. Inquiries receive attention no matter how much tension is generated by arguments, competing demands on the biller's attention, or approaching deadlines that cannot be met. Telephones are answered; the biller listens to the problem and either resolves it on the spot with a satisfactory explanation or promises to take action.

A possible alternative for the biller is to take the positive and relatively quick step of passing the inquiry on to someone else, usually another biller who handles the sales team specializing in the product in question. This usually happens when the billing problem requires knowledge of billing arrangements specific to that product or when it depends on past conversations between the customer and another biller.

When a biller records information or receives written information and does not refer the problem to someone else for solution, an inquiry backlog is created. Notes and memos regarding unresolved inquiries accumulate on the desk top and are often either stored by the biller in one or more in-baskets or consolidated into stacks of problems on the desk. If these documents come in faster than the inquiries can be resolved, they may move from desk top to drawer and form a backlog file.

Automatic backlogs are created when the biller receives a mass of inquiries simultaneously. There are several circumstances under which this can occur: A biller may inherit the accumulated inquiries of another biller who fails to return from a protracted absence;

inquiries concerning a large customer who uses many machines may arrive from regional headquarters; people in credit may communicate a list of problem cases that they have accumulated in trying to collect past-due bills.

Decision 2: When to work on resolving inquiries.

This decision becomes increasingly important as the inquiry backlog grows. A biller who has a large backlog often tries to schedule blocks of one to several hours to devote to inquiry resolution. If it is possible to finish working the reports that count toward the unbilled revenue target, an entire day may in theory be set aside for working inquiries; this means that the biller will attempt to resolve as many backlogged inquiries as possible while dealing with her other responsibilities.

If the backlog is slight, or if there is simply no foreseeable block of time for resolving inquiries, the biller may work on a case or two when she desires a change in activities.

Decision 3: Which cases to resolve first.

Inquiries vary with the dollar amount involved. There are actually two relevant dollar amounts: the total amount of the disputed bill, and the amount of the potential adjustment required. Billers tend to focus on the amount of money that was incorrectly billed, rather than on the total amount of the bill.

The relative importance of a customer is also a major consideration. If the customer has a large account and rents many machines correction of even a small billing problem may receive priority in order to maintain the customer's goodwill.

Another way of choosing among alternative cases for processing is to classify them according to difficulty. If a biller wants a challenge or if the case is pressing and time is available, she may choose a hard case. For a change of pace, to relieve pressure, or when only a small amount of time is available (for instance just before a meeting), the biller may work on some easy ones. Cases are classified according to (1) the size of the company involved, with larger companies typically presenting harder problems (such as multiple invoice pulling) and (2) the dollar amount, with large dollar amounts often requiring extensive discussion and managerial involvement. Cases may be classified as easy if they are routine billing problems or if they require the biller to leave her desk to consult with someone in another part of the building.

The extent to which other people in the branch are involved is a crucial consideration in picking cases that receive priority. Many inquiries are brought to the biller by salespersons who want to avoid losing a customer through confusion over billing. Salespersons tend to set case priorities according to size of machines instead of dollar amounts under dispute; this sometimes conflicts with the biller's evaluation of a particular case. When evaluating

alternatives, the amount of information recorded on a case can be a deciding consideration, with preference given to inquiries for which a good explanation of the problem is given, along with an appropriate presentation by the salesperson. When a salesperson demands impossibly rapid action, applies too much pressure, or blames the biller personally for the problem, the biller is likely to give priority to other cases.

Another very important consideration with regard to inquiries brought to the biller from sales is the history of the relationship between the biller and the individual salesperson. Billers regularly need certain information from salespersons, such as meter readings obtained from customers. Deadlines for receiving this information for computer input are fixed and foreseeable, and billers evaluate the timeliness, accuracy, and completeness of the information they receive from the salesperson, eventually forming an image of the salesperson based on these evaluations. Another factor contributing to this image is the frequency with which the salesperson attempts to persuade a biller to credit an account in order to placate a customer. Legitimate inquiries based on erroneous bills often take this form; but repeated use of this strategy can lead the biller to doubt that an account is being handled properly.

The billers receive many customer inquiries from the people in Credit and Corrections with whom they share coverage of a sales team. This credit/billing pathway tends to be used more often and more directly than that between billing and sales since the former groups are in the same room. Nevertheless, structural conflicts arise that people work out in the context of their daily decisions.

People in Credit and Corrections are targeted on the dollar amount they collect, and they tend to focus on the total amount of the unpaid bill rather than on the disputed portion of it. Because customers often dispute the correctness of unpaid bills, the credit representative is in a position to pass many inquiries on to the billers, and under pressure of targets will tend to favor the resolution of difficult problems by billing adjustments. Determining the relevant facts in such cases often involves research, such as checking customer files, records of payment, and past-due invoices, and billers and credit workers often have conflicting opinions about who is responsible for doing such research. One vehicle for expressing such opinions is the allocation of priorities to specific inquiry cases.

Decision 4: Where to file the completed adjustment package?

A copy of each adjustment package is stored in the branch for at least one year after the adjustment is processed. Although a central file cabinet exists for these packages, individual billers follow different procedures in filing them. Completed packages accumulate in billers' desk drawers for three reasons: (1) the account may be subject to further inquiry when the customer receives the updated invoice; (2) the transaction may be rejected by the computer system due to a technical error and thus may need to be resubmitted; (3) it is more convenient to accumulate a batch of packages and then to file

them in the central file all at once. One biller prefers to maintain fairly complete files of all adjustment packets and supporting documents and to extend her file space with cardboard boxes as necessary. The alternative is to distribute this material between files maintained for all individual machines and the central adjustment packet file, perhaps throwing away the remainder. This particular biller considers her system useful in contrast to more centralized filing, since folders are notoriously hard to retrieve from the central file cabinet.

As this analysis shows, the process of making decisions in the course of resolving customer inquiries is complex. It involves much more than the immediate billing problem. The customer assistant calls upon her extensive knowledge of branch organization, the history of various work relationships, and past experiences in order to determine a course of action for any given problem. In the course of completing an account resolution, the employee must take into consideration the demands of monthly targets, the need to obtain cooperation from others, and the behavior of external third parties -- the customers. As this summary has shown, decision-making in a complex environment such as this is far more complicated than a procedure manual might indicate.

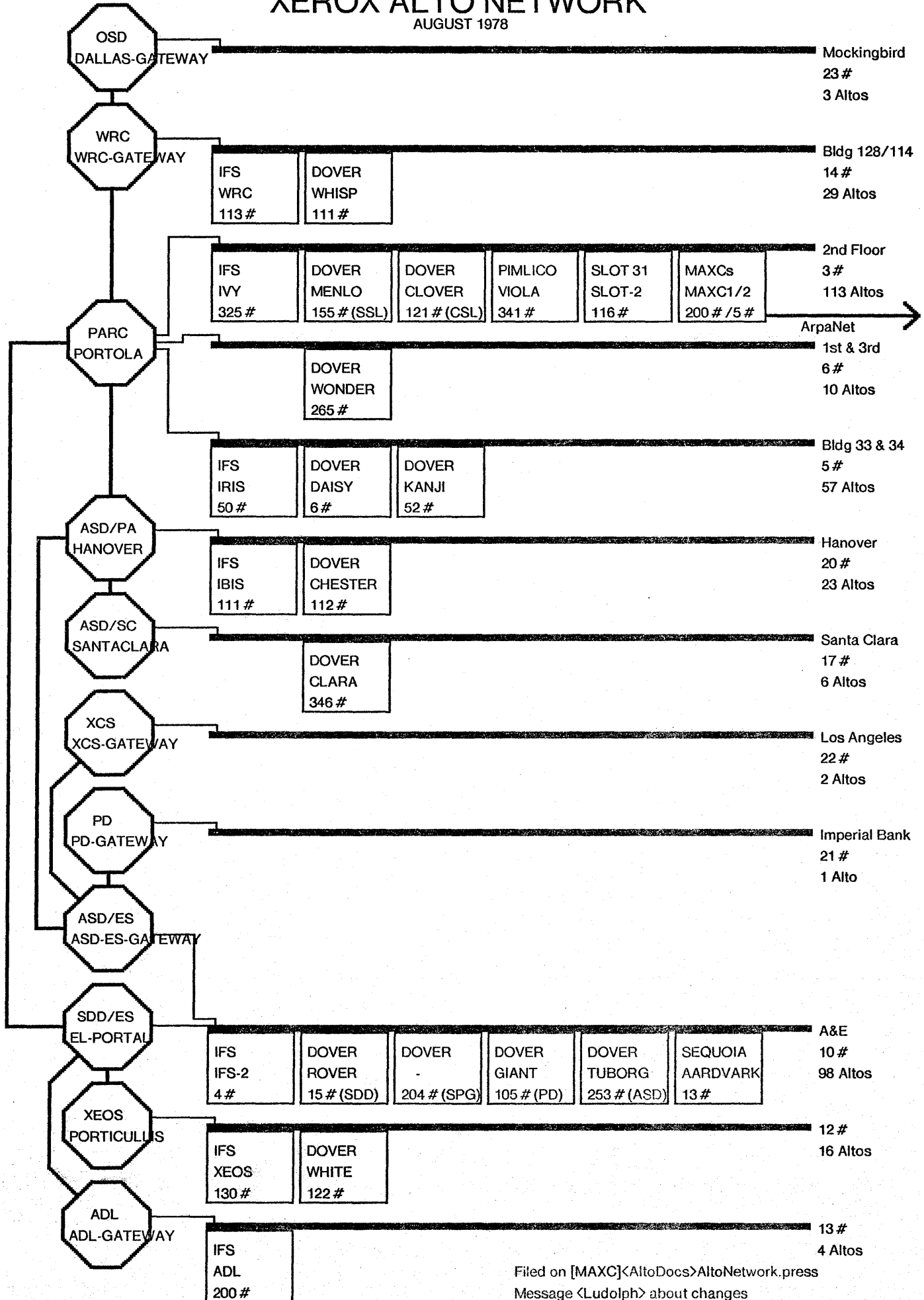
New office technologies should take into account this wide range of information, both social and job-related, that influences the decision-making process. Furthermore, it is important to realize that the choice of alternative strategies is dictated as much by the position and goals of individuals in the organization as it is by the demands of the task itself.

CONCLUSION

The observations and analyses presented in this report summarize our findings from the six-week field work period. They are intended primarily for the use of those involved in developing new types of office technology. It is vital to understand the working environment into which new forms of technology may be introduced. This project could not have been completed without the invaluable cooperation of the members of the three customer service divisions with whom the anthropologists worked. This summary is presented to them in order to indicate how their assistance has contributed to the better understanding of the world of office employees.

XEROX ALTO NETWORK

AUGUST 1978



Whole ALTO World Newsletter

Technology and Tools

XEROX

August 31, 1978

SPECIAL ANNOUNCEMENTS

WHOLE ALTO WORLD MEETING - The next meeting will be held Tuesday, October 17th, from 9 am to 3:30 pm at the Webster Research Center, Webster, New York. Jim Iverson, ISA, and Darwin Newton, Ginn, are our hosts. This is the first meeting to be held on the east coast, so come back and have a look around. For those of you unfamiliar with the area, suggested accommodations are the Depot in Webster, The Sheraton and Marriot at the airport (both have large indoor pools), and the Americana and Holiday Inn in Rochester.

WAW COORDINATOR MOVES TO NEW LOCATION - Frank Ludolph, the WAW coordinator, has moved to the Hanover building in Palo Alto. He can now be reached at Intelnet 8*923-4652. Mail should still be addressed to PARC. The move results from Frank's assumption of additional duties with User Support Services in ASD, a role complementary to that of coordinator.

GENERAL NOTES

WASHINGTON PROBE NEWS ITEMS - Several articles have appeared recently about the ASD Washington probe. They are in Datamation (Aug. pp. 54-55), Typeworld (Aug. p. 4), and the Wall Street Journal (Aug. 15 p. 4). (Copyright laws prohibit their inclusion here.) In general, they describe the equipment - not wholly accurate - and comment on the future of such systems. *These are not Xerox cleared releases so you should neither confirm nor expand upon their content.*

ALTO DLS/TELENET LINK - The Alto Data Line Scanner has recently been connected to the Telenet Corporation's digital data network. This connection, which supplements Intelnet or direct dialing, enables users with a terminal such as a TI 7xx to couple into the PUP network from the many places in the world where Telenet now offers services. (Local Palo Alto users should continue to use 493-3121 rather than the Telenet Tip in San Carlos.) The most typical use of the service is to access MAXC to send and receive mail.

ASD is graciously paying the fixed cost for this service (about \$700/mo). The connection charges (about \$7/hr) are to be paid by each user organization (based on organization name/password use) via a transfer agreement with PARC. Detailed usage, i.e. individual/time, of the Intelnet variety is not available. (Direct dial and Intelnet coast-to-coast connection costs \$20 to \$25/hr. and is of lower quality.)

If your organization would like to use this service, Ted Strollo suggests that you contact the indicated member of your organization: Jim Anderson, XCS; Ron Rider, PD; Carlos Santiago, GSD; Dick Sonderegger, SDD; Don Stewart, XEOS; or Chuck Thacker, ED. (ASD, PARC, and WRC are/will be users.)

NEW ALTO NETWORK ADMINISTRATOR - Tim Carroll of RTCC in Webster will soon be assuming control of the network. Tim is looking for a person to be the network wizard to handle both technical and administrative matters. This person will handle the network topology, Gateway boot file updates, name server directory maintenance, network troubleshooting, new site installation, and common carrier interfacing. The position is in Webster with heavy PARC travel initially. If you know of someone suitable, call Tim at 8*222-2100 or message <Carroll>.

Whole ALTO World Newsletter

SUBSCRIPTION RENEWAL INPUT - The Newsletter subscription renewal effort appears to be complete. Distribution now is about 165 copies, the same number we started with two months ago. Several renewal slips suggested additional topics which will be covered in future editions: want ads and requests (the new Marketplace section), activities of Alto users, and a guide to papers. Each of these topics will be covered or incorporated in future WAW publications.

NEWSLETTER PUBLICATION - Beginning with this issue, the Newsletter is being reproduced and distributed by GSD Office Services in El Segundo. This relieves the coordinator of personally printing, stapling, stuffing, and mailing each issue as he has done in the past. Requests to be added to the mailing list, problems, etc. should still be directed to Frank at Intelnet 8*923-4652.

MARKET PLACE

Market Place provides a forum for Alto users to make offerings and requests for Alto related hardware and software. To place an "ad", send the text to the coordinator, Frank Ludolph (PARC), message <Ludolph>, or phone Intelnet 8*923-4652.

BUFFERS AND HIGH SPEED ALTO INTERFACES - Dave Cronshaw, ADL, suspects that significant amounts of Alto related hardware and software are developed which are never made "public". In an effort to break some of it loose he offers a couple of PARC/ADL items, an input-output buffer and a 150Mbit parallel to serial/serial to parallel converter. The two might be coupled to produce a high speed buffered interface for the Alto. Descriptions and block diagrams are attached to the Newsletter.

WANTED, HARDWARE FOLKLORE AND A FILE LOCATOR - Dave Damouth, WRC, suggested that the Newsletter might be used to voice requests and offered two examples. Hardware designers could use a hardware folklore document similar to the software folklore documents that have been surfacing recently. It would be of help to those learning to design hardware interfaces to the Alto.

The second request is for a file locator. The proliferation of files and file servers has increased the difficulty of finding a specific file and lead to excessive replication of seldom used files. He suggests an automatic periodic survey of all files on all servers (perhaps using BROWNIE as a base) and a "Find file" command in IFS.

TOOLS

MAINTENANCE NOTES

VACUUM CLEANERS - Noise spikes from motor driven electrical equipment, such as vacuum cleaners, can affect Altos on the same AC circuit. The spikes won't damage the Alto itself, but they may cause running programs to do funny things, e.g. SWAT or write garbage on your disk. If the machine must be left running overnight (other than DMT or the like) post a note telling the janitor not to vacuum your office.

APPARENT (FALSE) POWER SUPPLY FAILURES - Dropping and resuming power, e.g. power failures or thrown circuit breakers, to a powered up Alto can cause its power supplies to reset, with the appearance of a power supply failure (lose of lights on the disk drive). The supplies are supposed to reset when the Alto is turned off and on, however, they generally don't respond immediately. Wait about ten minutes after turning the machine off before turning it back on. If there are still no disk lights, call a maintenance person.

Whole ALTO World Newsletter

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local IFS server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [XEOS].

NEW RELEASE: DYNLDR - This package is used to customize a program at runtime by dynamically loading a BCPL .br file. Documentation on its use will be available soon. Retrieve <Alto>DYNLDR.br

NEW RELEASE: POPPY - Poppy is an on-line documentation system that supports indexing and automated definition look-up. *It is a MESA 4.0 program so it should not be loaded onto a disk containing LAUREL 1.x.* The documentation is self-contained; just start up POPPY and bug HELP. Retrieve and execute [IVY]<Poppy>GetPoppyImageEtc.cm.

New Documentation

BCA/MLDR - Basic Cross-Assembler is a general purpose microcomputer assembler. To date it has been used to assemble code for several microcomputers: Intel 4040, 8080, 8748 and 8086, Motorola 6800, MOS Technology 6502, Texas Instruments TMS100, and Zilog Z-80. It takes about a day's work to customize BCA for a new machine with an 8 bit instruction and only a few unusual features. MLDR combines several BCA produced files to produce one standard Micro format binary file. Retrieve <AltoDocs>BCA.press.

FANCYTEMPLATE - This package is an enhanced version to the TEMPLATE package that formats output to a stream according to a template. The enhancements include hexadecimal output and a provision for a user-supplied routine to format non-standard data types. See <AltoDocs>FANCYTEMPLATE.tty.

MICRO - MICRO is a machine-independent microassembler originally developed for Maxc1 and since used for Maxc2, Dorado, and D0 as well as for several smaller projects. It does have a specific target machine; rather it has a general facility for defining fields and memories, a standard string-oriented macro capability, and a parsing algorithm that allows setting fields in memory. The document is of interest primarily to someone who will define a new assembly language for a machine. Retrieve <AltoDocs>MICRO.press.

MESA TOOLS GUIDE - The Tools Environment Guide for Uool Users is now available on [IRIS]<Tools>Alpha>Documentation>ToolsUserGuide1.press and ToolsUserGuide2.press.

ReReleases - Subsystems

ANALYSE - In addition to bug fixes one new feature has been added; when using the "titleblock" facilities, Sil drawings may now be included for use as reference and not be analysed. retrieve [IVY]<Sil>ANALYSE.run. Updated documentation is in the SIL Manual.

BUILD - The new version invokes the hardcopy feature in SIL rather than NPPR. Retrieve [IVY]<Sil>BUILD.run. The documentation in the SIL Manual has not been updated.

Whole ALTO World Newsletter

HARDCOPY - In addition to retrieving files from a server and invoking BRAVO to hardcopy them, HARDCOPY will now also EMPRESS press files. Both the .run file and the User.cm slice have changed. Load <Alto>HARDCOPY.dm and install in accordance with the updated documentation <AltoDocs>HARDCOPY.tty.

MICRO - This release fixes all known glitches and improves throughput by 20-25%. Retrieve <Alto>MICRO.run. A new manual is on <AltoDocs>MICRO.press.

PNEW/PROM - PROM now has the same switches as PNEW. Additional proms have been added to PNEW. Old command lines still function properly. Retrieve <Alto> PROM.run and/or PNEW.run. Revised documentation for these prom fusing programs is on <AltoDocs>PromManual.run.

SIL - SIL has had both a major and bug fix releases since the last Newsletter. New features include a hardcopy mode (similar in concept to BRAVO's Look Hardcopy), font faces (bold and italic), and direct hardcopy output. NGPR and NPPR are now obsolete. Retrieve [IVY]<Sil>SIL.run (a [MAXC]<Sil> copy has been requested). The revised documentation is on [MAXC]<Sil>SILManual.press.

ReReleases - Packages

MESALIB - MESA 3. items are now in <OldMesaLib> on IRIS and ISIS. <MesaLib> contains MESA 4. packages. Summary.press contains thumbnail descriptions of the available packages.

TECHNOLOGY

A natural use of OIS is the storage and retrieval of personal information, i.e. information primarily of interest to the person who puts it in. Richard Sauvain, WRC, has had an interest in and been a user of such systems for a number of years. His paper, Computer-Based Personal Retrieval Systems, provides some background information and a topically organized, annotated bibliography on this subject.

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC]<AltoDocs>WawNewsM-YY.press or may be obtained from the editor, Frank Ludolph, ASD, by messaging <Ludolph> or calling Intelnet 8*923-4652.

Computer-based Personal Retrieval Systems

by

Richard W. Sauvain

Xerox Corporation
Webster Research Center
Information Sciences Area

August 24, 1978

Abstract: This is an annotated bibliography on "personal" information storage and retrieval systems. It catalogs work done to date on computer based retrieval systems in which the input is selected or indexed by the end user of the information. The bibliography is preceded by a short commentary defining further the meaning of "personal", describing the history of the area, and taking a brief look at the state-of-the-art and currently important research areas.

DRAFT

This is to be issued as internal report X78-02533; I also plan to be submit it for outside publication. Critical comment is welcome; I can be reached at IntelNet 8*222*5298, or at account ISA with SndMsg!

Introduction

Techniques for organizing one's personal store of written material have long been of interest to information workers. Almost everyone doing research, record keeping, or analytical work builds a collection of notes, records, letters, memoranda, papers, and the like. In time, such collections get sufficiently large that it is hard to retrieve information. This motivates development of some sort of organization scheme to facilitate retrieval. Retrieval effectiveness tends to fluctuate, improving with the addition of new techniques, and worsening as the collection grows.

Computers have been used to facilitate retrieval in personal collections roughly since the mid-60's, when computer resources began to get cheap enough to make personal applications feasible. Work accelerated with the spread of time-sharing and mini-computers, and the recent rapid increase in availability of microcomputer-based personal computers promises to bring a new surge of interest in personal information retrieval.

This paper catalogs work done to date on computer-based personal information retrieval systems. These are defined as retrieval systems in which the input is selected, keystroked, or indexed by the end user of the information, and in which the collection is organized to match the individual's own approach to the subject matter. Note that this definition does not cover large general purpose retrieval systems (e.g. commercial search software or large library systems), unless they are used for a personal application. A few papers on manual systems and indexing practice are included for completeness. General purpose retrieval software operating on micro-computers *is* covered (since it is likely to be used to develop personal systems).

Some Notes on the Restriction to *Personal* Systems

The term "personal" connotes (in this paper) that the the end user of the information is the same person as the one who puts it in. Though generally single user, personal databases are occasionally shared by a small group of people who share a common conceptual framework. This 'putting in' activity may be regarded, more concretely, as imposition of some kind of retrieval-facilitating structure on a specialized body of information -- of moderate size (generally not more than one or two megabytes).

An important aspect of personal retrieval systems is that the end user generally does not want to rely upon anyone else's index terms. The motivation for building a personal database is usually that personal indexing or organization *must* be performed to make the stored information truly useful. In other words, a large part of the usefulness of a personal database is that the information has been evaluated by the collector relative to a personal conceptual framework. This is in contrast to large, 'public' retrieval systems, where the diverse nature of the audience makes it necessary to rely on 'general purpose' indexing.

Personal systems are of interest for at least three reasons. First, most information gathering - especially in scientific or analytic work - is done in one's own collection. Libraries and even colleagues are used far less often. Secondly, these systems pose interesting requirements for accommodation to personal conceptual frameworks, for customization, and for accessibility - they have to be available very quickly when the need for the information is at hand. Thirdly, the age of personal hardware is upon us -- software to make good use of it is urgently needed.

Roots

Computer-assisted personal information retrieval systems have their roots in many areas: manual systems, simple reference listing programs, retrieval facilities in document preparation systems, large bibliographic retrieval systems, and the more general area of 'information work'.

Most people approach information organization and retrieval by setting up some sort of manual system. This might be a filing cabinet - perhaps ordered by author name, or a subject file of 3 x 5 cards, or a more elaborate systems involving edge-notched cards with coincidence of holes indicating subject classes. There is a great body of advice available on how to set up manual filing systems, and on how to index their entries (see for example [Jahoda 1970]). Note: throughout this paper, references formed from the first author's last name and the year of publication will be used. The intent is not to ignore co-authors, but rather to provide a brief mnemonic, reference symbol.

The first substantial use of computers in the personal database area was to maintain lists of bibliographic references. [Burton 1969, Heaps 1968, Smith 1969, Wallace 1966, Yerke 1969, and Yerke 1970] are examples. An 'information worker' would typically have author, title, location, and some form of subject descriptors keypunched for references in some specialized area of knowledge. The computer would then be used to produce listings sorted by author or subject, or in some cases to produce KWOC indices from the titles. Some minimal editing or file updating facilities were available.

At about the same time, computers began to be applied to document preparation and text analysis. Pre-cursors of modern word-processing systems began to deal with the problems of filing and retrieving documents under preparation. This included the idea of using the computer to assist in the indexing process. [Bobrow 1969, Carmody 1969, Nelson 1967, Walker 1967] report early work dealing with retrieval aspects of text preparation.

Another root is in programs that provide time-sharing access to large bibliographic databases. In [Kessler 1967], for example, the possibility of individuals wanting to extract and maintain personal copies of parts of very large general-use databases was recognized. In addition, these larger systems have had an influence on software for personal applications.

Lastly, much important work in this field derives from the study of a more general problem: how to effectively support an individual's information processing work, in the broad sense. [Bush 1945] published probably the earliest vision of computer assistance for information work. His idea of a "memex" machine is heavily cited. [Englebart 1961] developed some of the psychological and human factors aspects of information handling work, while [Nelson 1965] did some similar thinking, though oriented more towards support of creative writing.

Comments on the State of the Art

Judging from the significant number of books and papers published, and from the fact that sections on 'support for personal files' are beginning to appear in survey articles, the field of personal retrieval systems is at least in its adolescence. A large number of people have tried them out, and have found them useful and beneficial. Computer-based systems have been in use for a reasonable amount of time (since about 1966).

Many such systems use unstructured 'flat files' of bibliographic-style records. More elaborate data structures (indices, linked lists, hierarchies) have been explored and found useful for some applications, particularly when on-line retrieval is desired.

Query and reporting facilities available in personal systems range from simple 'list the whole database' commands to more elaborate boolean queries, sorting, and facilities for full text search.

Application of general-purpose database management software to personal collections is commonplace. To an increasing degree, these systems provide the ability to 'customize' data structures and the user interface to particular applications.

Cost-effective personal retrieval systems have been developed for large time-shared computers, and for mini-computers. The micro-computer era is just beginning, with at least two interesting retrieval software packages available [Gerritsen 1978, Morrill 1978], and a surge of activity in this area predicted. With workable systems costing a few thousand dollars, the cost-effectiveness of personal retrieval work is improving dramatically.

Some thoughts on areas for further research

Portable systems . Cheap, portable computer systems with enough mass storage (one or two megabytes) to do interesting retrieval work are not far off. Hardware costing under a thousand dollars will probably be ready with a few years; software to make good use of it still needs to be developed. A related area is that of connecting portable computers to large central retrieval systems -- to use as terminals for broad searches, or transfer to the small system a subset of retrieved items for further processing (such as personalized subject indexing).

Microform/digital hybrids. Ever since Bush's 1945 paper on the memex idea, the notion of a system that allows convenient input of imaginal data (in compact, quickly retrievable form) has been attractive. Photographs, long hand notes and sketches, signatures, etc. could be stored. Storage densities should be much higher than the current digital technology commonly used. The problems come in producing digital indexes (or other retrieval facilitating structures) coordinated with the microfilm, microfiche, holographic store, or whatever. Another stumbling block is lack of convenient ways to annotate or change imaginal data.

Association networks . Adding retrieval facilitating structure to data can be viewed as imposing a personal conceptual viewpoint (or set of associations) on the data. Bush visualized elaborate associative trails through personal databases, and indeed a whole new profession of 'trail-blazers'. If an association between two items is noticed, (in browsing through a database), it is important to be able to note it down, to have it there for later use. Convenient ways to make and follow associations are non-trivial to provide.

Human factors studies. A primary reason that people use personal collections is convenience. If personal retrieval systems are not easy to use, accessible when they're needed, and easily customized to one's own way of working, they will not be used. A great deal of experimentation with techniques to provide these attributes is in order. Detailed studies of the human interface are needed.

Guide To The Bibliography

The following is a brief taxonomy of personal retrieval systems, intended as a guide into the annotated bibliography that follows. The intent is to provide a few pointers for those with specific interests.

Sequential file, oriented towards printed index production

[Bridges 1970]
[Burton 1969]
[Hart 1975]
[Hatcher 1976]
[Purdy 1971]
[Smith 1969]
[Wallace 1966]
[Yerke 1969]
[Yerke 1970]

Relationship of personal systems to libraries

[Burton 1972]
[Burton 1973]
[Korfhage 1978]

Systems using microform image storage

[Bush 1945]
[Hoekstra 1977]
[Johnston 1975]

Implementation tools

[Gerritsen 1978]
[Mittman 1975]
[Reynolds 1976]
[Tharp 1978]

Human factors studies of personal information work

[Dean 1963]
[Englebart 1961]
[Englebart 1973]
[Linn 1972]
[Sauvain 1970]

Applications of generalized DBMS's to personal databases

[Hatcher 1976]
[Kessler 1967]
[MacKenzie 1977]
[Mittman 1975]
[Szonyi 1974]

Retrieval facilities in document preparation systems

[Carmody 1969]
 [Englebart 1973]
 [Nelson 1965]
 [Nelson 1967]
 [Robinson 1973]

The personal, portable computer idea

[Hubbard 1976]
 [Kay 1977]
 [Korfhage 1978]
 [Lees 1977]
 [Van Ree 1976]
 [Warren 1977]

Statistical studies of manual systems

[Burton 1972]
 [Jahoda 1966]

Other overviews

[Lancaster 1973]
 [McEowen 1969]

Advice on setting up manual systems

[Jahoda 1970]
 [Loring 1971]
 [Norton 1970]

Interactive, retrieval oriented systems (including full text retrieval)

[Glantz 1970]
 [Heaps 1968]
 [Jones 1977]
 [Leggate 1977]
 [MacKenzie 1977]
 [Mittman 1975]
 [Reitman 1969]
 [Reynolds 1976]
 [Robinson 1973]
 [Saunders 1977]
 [Sterner 1976]

Systems allowing inter-item structure

[Bush 1945]
 [Carmody 1969]
 [Cashin 1974]
 [Englebart 1973]
 [McEowen 1969]
 [Nelson 1965]
 [Reitman 1969]
 [Robinson 1973]
 [Sauvain 1970]

Intelligent Front Ends

[Linn 1971]
[Linn 1972]
[Walker 1967]

User-assisted indexing for personal retrieval

[Bobrow 1969]
[McEowen 1969]
[Walker 1967]

Systems implemented on mini or micro computers

[Garfield 1978]
[Gerritsen 1978]
[Hoekstra 1975]
[Johnston 1975]
[Kay 1977]
[Morrill 1978]
[Sargent 1978]
[Saunders 1977]
[Sterner 1976]
[Van Ree 1976]
[Warren 1977]

Annotated Bibliography

[Bobrow 1969]

Bobrow, D.G.; Kain, R. Y.; Raphael, B.; Licklider, J.C.L., "A Computer-Program System to Facilitate the Study of Technical Documents", *American Documentation*, Vol. 17, p.186, October 1969.

This paper reports on a system called SYMBIONT, which uses a dedicated PDP-1 with a 10 inch square CRT and a lightpen. The user page-turns through a body of text, and can designate (by typing in or selecting on screen) tags, which go into a displayable glossary and can be used in retrieving. The authors view the retrieval problem in small document collections as primarily one of passage retrieval.

[Bridges 1970]

Bridges, Kent W., "Automatic Indexing of Personal Bibliographies", *Bioscience*, Vol. 20 No. 2, pp. 94-97, January 1970.

An index preparation program written in PL/1 for the IBM/360 is described. The user keypunches bibliographic references and keywords; the program produces a bibliography, indices by author and source, and a KWIC index of the titles. A frequency distribution by year of publication is also produced (used for things such as finding years that were neglected in compiling a bibliography).

[Burton 1969]

Burton, Hilary D. and Yerke, Theodor B., "FAMULUS: A Computer-Based System for Augmenting Personal Documentation Efforts", *Proceedings American Society for Information Science*, Vol 6, pp. 53-56, 1969.

FAMULUS is a collection of programs designed to assist a working scientist in organizing a personal library. The system is based on eight programs which operate in batch mode, producing files on magnetic tape and reports on a line printer. File building, editing, indexing, and retrieval facilities are present, operating on file definitions adapted by users to their individual needs. Some interesting examples of actual databases (primarily of bibliographic references) are described. An attempt was made to make the system input simple enough to be used by a non-programmer.

[Burton 1972]

Burton, Hilary D., "Personal Documentation Methods and Practices, with Analysis of Their Relation to Formal Bibliographic Systems and Theory", Ph.D. Thesis, University of California at Berkley, December 1972.

This is a study of 13 FAMULUS databases - primarily of journal article references in the area of natural resources research - to determine their structure and composition, and their relationship to other information services. Attempts were made to determine the amount of overlap between the personal files and widely used formal secondary information sources, such as *Biological Abstracts*. Common features noticed were: low overlap with formal sources; heavy use of informal information channels, concentration of most of the database in core sources, emphasis on recent information (within 5 years), and use of personalized control vocabularies.

[Burton 1973]

Burton, Hilary D., "Personal Information Systems: Implications for Libraries", *Special Libraries*, Vol. 64, pp. 7-11, January 1973.

This paper is primarily about the effects of personal systems on formal library services (including the possibility that people will cease to use the library if they have a good personal system). Conclusions are that personal information systems are not "personalized libraries" (there is high reliance on informal information channels - such as colleagues; control vocabulary is highly personalized,...); formal library services should concentrate on developing complementary programs, such as providing full text copies of articles, and looking into supplying machine readable catalog data.

[Bush 1945]

Bush, Vannevar, "As We May Think", *Atlantic Monthly* Vol. 176, No. 1, pp.101-108, July 1945.

In this seminal article, Bush visualized a desk-top personal documentation system known as 'Memex'. He described it as a mechanized private file and library, based on microfilm. Noting that the human mind works by association, Bush set forth the idea that selection of relevant material from a personal collection is often done better by *association* than by indexing. Memex users would be able to tie items together into named 'trails', forming branching chains of related items as they browsed in their collections, and could add commentary as they worked. Bush also predicted that most of the memex contents would be purchased on microfilm, ready for insertion into the device: books, pictures, periodicals, correspondence, etc. The memex would also contain a platen for input of arbitrary material to microfilm storage.

[Carmody 1969]

Carmody, Steven; Gross, Walter; Nelson, Theodor H.; Rice, David; and Van Dam, Andries, "A HYPERTEXT Editing System for the /360", Report from the Center for Computer & Information Sciences, Brown University, March 1969.

HES (Hypertext Editing System) is an experimental text handling system implemented on an IBM 360/50 with a 2250 display and a private disk pack. Information is viewed as 'hypertext' - non-sequential writing. Information retrieval is present primarily to locate places in a document under preparation - in this sense it is retrieval within one's own personal document database. The system seems motivated by many of the ideas in MEMEX. It is also related to the SRI NLS system, but without that system's constraint to hierarchical files. A notable feature is the presence at the end of an area of text of a set of 'branches' - links to other parts of the document. The motivation here seems to be facilitating the writing of a CAI-type text. The general motivation is to "...allow the user to make reference to his work conceptually, by section or by context of ideas, however he feels is natural".

[Cashin 1974]

Cashin, P. M., Robinson, M. G., and Yates, D.M., "Experience with SCRAPBOOK, a Non-formatted Data Base System.", *Information Processing 1974-Proceedings of IFIP Congress*, pp. 1012-1016, 1974.

This is a summary of the applications of SCRAPBOOK, "an environment for people to use for their day-to-day office activities". This system is written in BCPL, and runs on a CTL Modular One processor via a packet switched network at the National Physical Laboratory in England. Information is stored as 'frames' (a frame is one screen full), and a set of frames may be grouped together as a record. Records may be named, and the end of a frame may contain a set of links to other places in the record-frame space - a link facility similar to NLS. They mention (but do not describe in any detail) applications of SCRAPBOOK to report writing, secretarial tasks, project management, software production, retrieval and update of frequently changing information, and on-line notetaking and access to documentation during meetings.

[Dean 1963]

Dean, Robert L., "It's Right Here Somewhere", *America*, Vol.108 pp. 404-405, March 1963.

An entertaining spoof on the methods we really use to locate items in our own personal collections! Points up some of the problems personal retrieval systems are designed to solve.

[Englebart 1961]

Englebart, D. C. , "Special Considerations of the Individual as a User, Generator, and Retriever of Information", *American Documentation*, Vol. 12 No. 2, pp. 121-125, April 1961.

An early and much cited paper arguing for the development of a separate discipline dealing with support of the information handling activities of individuals. The main problem with information handling work is short-term memory limitations - a person can work with only a few concepts at a time. To work with more than a few, there is a need to store and order them on an external medium. The author describes a personal note-taking system based on edge-notched cards. He notes that he found it very productive to "browse through a note deck, integrating what I find and what is stimulated in my mind, and synthesizing new concepts, considerations, and ideas on new note cards." The promise of computer support of such activity is anticipated.

[Englebart 1973]

Englebart, D. C, Watson, R.W., and Norton, J. C., "The Augmented Knowledge Workshop", *AFIPS Conference Proceedings*, Vol. 42 (1973 National Computer Conference), p. 9-21, June 1973.

Contains a good explication of "knowledge work" - the broad concept of which personal IS&R is a part. The work described - done at Stanford Research Institute - centers on an augmented knowledge workshop embedded in the ARPANET. Several computer-based knowledge tools are described, running on a PDP-10 under TENEX. Studying (composing, viewing, editing) on-line documents has long been a focus of this group. They work primarily with hierarchically structured text files called NLS files; the user's information space is viewed as a network of such files. There are some interesting

facilities for jumping around in the information space, and for viewing it in different ways. Also mentioned is an NLS Journal, made up of documents and messages submitted by ARPANET users. Automatic catalog generation processes produce author, catalog number, and title word indices, which are accessible on-line.

[Garfield 1978]

Garfield, Eugene, "Introducing PRIMATE TM - Personal Retrieval of Information by Microcomputer And Terminal Ensemble", *Current Contents*, Number 29 (July 17), p. 5-9, 1978.

This short article, one of a series titled "Current Comments", is 'an attempt to do some market research' on the idea of microcomputer-based personal information retrieval. The author is president of the *Institute for Scientific Information (ISI)*, which publishes *Current Contents* and provides several other information dissemination services. A. E. Cawkell, ISI's Director of Research, is working on a microcomputer based system to help manage, among other things, files of reprints obtained from information dissemination services. The system envisioned would consist of a small microcomputer, CRT terminal, and dual floppy discs. It would be able to store up to 10,000 subject-indexed bibliographic references. The article points out the possibility of selective dissemination of references via a floppy disc: the PRIMATE user would receive a disk of items matching an interest profile, and would scan it on the CRT, selecting those of interest and adding personalized subject indexing.

[Gerritsen 1978]

Gerritsen, Rob; Hackathorne, Richard D., "Micro-SEED Reference Manual", Available from: International Database Systems, 3700 Market St., Philadelphia, Pa. 19104, 1978.

Micro-SEED is a proprietary micro-computer version of the SEED database management package. This is a FORTRAN-based CODASYL database system (see the April 1971 CODASYL Data Base Task Group Report), providing full database management capabilities with network-type data structures. Machine requirements are a Z80-based microcomputer with 60K bytes of RAM and dual floppy disk drives. The CP/M operating system is used. One of the intended markets is for "software OEM" database applications; it would probably be most useful for highly structured and inter-related data.

[Glantz 1970]

Glantz, Richard S., "SHOEBOX: A Personal File Handling System for Textual Data", *AFIPS Conference Proceedings*, Vol. 37, pp. 535-546 (1970 Fall Joint Computer Conference), April 1970.

SHOEBOX is a system written in a LISP derivative for an IBM 360/50, using a private 2316 disk pack and 2260 display terminals. It is designed to model a personal desk file drawer; an example is a file of current affairs information used by a news analyst. A 'file drawer' is made up of 'files', which are sequences of 72 character lines, which may be grouped into items. Lines may also have a one or two character category code (to indicate author, date, ...). Access to the information is via file name & line number, or by textual patterns. Good attention has been paid to human factors (e.g. users can rename commands, there are string editing facilities, and one can perform a search on the results of a previous search). Text searching is quite elaborate: right and left truncation are allowed, you can look for occurrences of words within same sentence, within n words, and so on. SHOEBOX also contains a search pattern definition command - useful

in building up complex searches. It also has reorganization facilities, which are useful for 'cut and paste' activity.

[Hart 1975]

Hart, Sheila L.; Parker, K; Price, J.A., "CAPRI- A Computer Aided Personal Reference Index System for Use by Individual Research Workers and Groups", AWRE Report No. 02/75, United Kingdom Atomic Energy Authority, Aldermaston, England , March, 1975.

CAPRI is a Computer Aided Personal Reference Index system, designed for individual research workers and groups having access to medium sized computers such as the IBM 360/30. The system is based on a subject index called CINDA (Computer Index of Nuclear Data). Input is in the form of punched cards, files are kept on magnetic tape, and the output is bibliographies ordered by date and by subject. A fixed bibliographic format is used, though there is provision for short notes. The report includes flowcharts and FORTRAN source code for the software, and full operating instructions (including JCL for the IBM 370/168).

[Hatcher 1976]

Hatcher, Myron E. et. al., "INFORMATION RETRIEVAL: Program for personalized article retrieval system", *Behavior Research Methods & Instrumentation*, Vol. 8 No. 6, p. 514, December 1976.

A batch-oriented system, strictly for bibliographic references, is described. Descriptors (either topics or sub-topics) are assigned to each article, and output of the system is a list of articles that have a set of descriptors. The programs are written in COBOL for the Xerox Sigma 6; they utilize a DMS database

[Heaps 1968]

Heaps, Doreen M. and Sorenson, Paul, "An On-Line Personal Documentation System ", *Proceedings American Society for Information Science*, Vol 5, pp. 201-207, 1968.

This paper reports on a pilot project to provide a small database of bibliographic references, augmented by notes and keywords. The software, written in APL, lets the user display, search by keywords, or add to items. Database size was limited to around 230 items. Intended use was to support a teacher making up lists of references for courses. The query language was found to be unsatisfactory; developing a good query language was noted as a significant problem.

[Hoekstra 1977]

Hoekstra, W.; Evers, V.H.C., "Automated Microfiche Filing System for Personal Use", *IEEE Transactions on Professional Communication*, Vol PC-20, No. 4, pp. 228-233, December 1977.

Described in this paper is a personal-use computer-based microfiche filing and retrieval system, oriented towards helping a manager/secretary team keep track of daily correspondence. Incoming mail is filmed on microfiche and indexed by author, date, and keywords such as business entities and tasks. A document database is maintained on a mini-computer. Searching is by author names, time windows, and keywords in any combination. A search results in a list of title-like document descriptions, from which individual documents can be selected -- these are displayed in full from a microfiche

carousel at the terminal. Retrieval is fast enough (a few seconds) to allow casual browsing, and the effort threshold to look at a document is significantly lower than for conventional paper files. In this personal application, indexing is seldom based on document content; the more important index terms are related to the activities of the user of the database. Field experience with a real manager and a 4000 document database is described. One interesting result: 65% of the queries could not have been answered easily via conventional file folder headings.

[Hubbard 1976]

Hubbard, H. P., "Personal Computer", *IBM Technical Disclosure Bulletin*, Vol. 19, No. 7, pp. 2419-2423, December 1976.

This is a proposal for a hand-held personal calendar device to assist in organizing and using one's time. The main function is to remind you of appointments; though it can also be used for note-taking, shopping lists, keeping addresses and phone numbers, etc. The users keys in a date and time, and an associated 26 character reminder. When that time arrives, the device beeps and displays the reminder. There is also provision for scrolling through the messages; a primitive retrieval facility! The device would use a 40 character liquid crystal display with 7 fields, a full alphanumeric keyboard, and magnetic bubble memory sufficient to hold 500 40 character 'messages'.

[Jahoda 1966]

Jahoda, G., Hutchins, R. D., and Galford, R. R., "Characteristics and Use of Personal Indexes Maintained by Scientists and Engineers in one University", *American Documentation*, Vol. 17 No. 2, pp. 71-75, April, 1966.

Presents results of a survey of graduate school faculty members who maintain personal indexes ("organized collections of documents and/or homemade references to documents that the researcher keeps in his office"). The authors attempted to find out size of these collections, rate of growth, frequency of use, physical form, and other important characteristics. Somewhat over half of the faculty members (who were in science and engineering departments) kept personal indexes. About 78% of the indexes contained fewer than 3000 documents. Most of them grew at 30 or fewer documents per month. Information on filing and access strategies is also given.

[Jahoda 1970]

Jahoda, Gerald, *Information Storage and Retrieval Systems for Individual Researchers*, Wiley-Interscience, New York, 1970.

Though the title seems to imply otherwise to computer-oriented readers, this is primarily a library scientist's advice on setting up manual retrieval systems. The principle topic of the book is indexing practice. Use of computer-prepared KWOC indices is discussed. A concluding chapter, "IS&R Systems of the Future", speculates on hardware support and man-machine communication features needed.

[Johnston 1975]

Johnston, R.J.D., "An Experimental On-line Retrieval System using Ultrafiche", *Program*, Vol. 9 No. 2, pp. 56-63, April 1975.

An experimental single-user system, using ultrafiche for display of retrieved results, is described. A single ultrafiche, containing 5890 references (with abstracts) from *Computer and Control Abstracts* was used in a Marconi Automated Microform Terminal, controlled by a PDP/8L. The PDP/8 also controlled a magnetic tape drive, on which were stored search data (index terms, author names, date, etc.) along with frame numbers. Sequential boolean searches were done on the tape, with results displayed as frames from the ultrafiche.

[Jones 1977]

Jones, A. U., "A User-Oriented Data-Base Retrieval System", *IBM Systems Journal*, No. 1, pp. 4-17, 1977.

An experimental database system based on APL is discussed. It was developed for "users who do not require the sophisticated resources of large database systems but do require many of the capabilities that they provide". The function and design attributes of the system are described including the reasons for basing the system on APL functions. The user of this system defines records made up of fixed length fields. All data is stored as text. As might be expected, elegant facilities for computing arithmetic functions of retrieved data are present. The system allows on-line database definition, data entry, and reporting.

[Kay 1977]

Kay, Alan C., "Microelectronics and the Personal Computer", *Scientific American*, Vol. 237, No. 3, pp. 230-243, September 1977.

While this article mentions retrieval systems only peripherally, it does provide an excellent look at the implications of the personal, portable, computers of the future. The author describes a desk-sized precursor to such a system, currently in use at the Xerox Palo Alto Research Center. This computer system has a microprogrammed CPU, a high resolution raster scan CRT, 3 megabyte disk, and pointing device called a mouse. He goes on to describe a simulation language called SMALLTALK, which is based on the notion of sending messages to objects. The language has a very flexible and powerful display interface. There is a lengthy discussion of teaching problem solving skills to children using SMALLTALK programming as a medium. Many fascinating illustrations of animation, editing, and simulation are given. The paper concludes with some remarks on the social implications of simulation and personal computers.

[Kessler 1967]

Kessler, M. M., "The "On-Line" Technical Information System at M.I.T. - Project TIP", *IEEE International Convention Record*, 1967, part 10, p. 40-43, 1967.

This early paper mentions possible use of the TIP software (one of the early on-line reference retrieval systems, used to search a database of 60,000 references in physics journal literature) for personal files. The example used is a departmental personnel file. The article notes that user control over the output format and good security features will be needed for personal applications.

[Korfhage 1978]

Korfhage, Robert R., "The Impact of Personal Computers on Library-Based Information Systems", *ACM SIGIR Forum*, Vol. 12 No. 4, pp. 10-13, March 1978.

A short paper predicting the availability of a hand-held intelligent terminal that can be used to access any public computing resource, for example, remote access to a library card catalogs, having a personal bibliography prepared and transferred to your personal computer. There is some speculation on the future availability of machine readable full text. The author thinks that a large proportion of future library usage will occur through its computer systems.

[Lancaster 1973]

Lancaster, F. W and Fayen, E. G., *Information Retrieval On-Line*, John Wiley & Sons - Melville Publishing Company, 1973.

Chapter 13 of this book, titled "On-Line Support for Personal Files", contains an excellent survey of personal retrieval system work. It gives an overview of the field, noting that the building of personal document collections is an essential ingredient of research and analytic activities. Pros and cons of maintaining personal collections are discussed at length. The chapter contains brief descriptions of several interesting systems.

[Lees 1977]

Lees, John, "The World In Your Own Notebook", *Creative Computing*, May/June 1977, pp. 54-56, May 1977.

This article from the personal computing literature provides a visionary description of computing equipment we may have in the future. Most of the discussion centers on the DYNABOOK concept, from the Learning Research Group of the XEROX Palo Alto Research Center. One of the most important applications of the DYNABOOK - a notebook sized portable personal computer - may be retrieval of "reference material, poems, letters, recipes, records, drawings, animations, musical scores, waveforms, dynamic simulations, and anything else you would like to remember and change".

[Leggate 1977]

Leggate, P.; Eaglestone, B. M.; Jarman, R. M.; Norgett, M.M.; Williams, A.P., "An On-line System for Handling Personal Data Bases on a PDP 11/20 Minicomputer", *ASLIB Proceedings* (GB) Vol. 29 No. 2 pp. 56-66, February 1977.

This paper reports on a sequential file system done in a macro-assembler language on a PDP/11-20 with a 2.5 megabyte disk, twin DECTapes, and TI 733 terminals. Users put in free-format text items, may also add tags (like <A> for author) to delimit data fields. Boolean type queries are allowed; response time is fairly slow. There is provision for editing items. They also report on usage experience during 1974 and 1975.

[Linn 1971]

Linn, William E., Jr., and Reitman, Walter R., "Referential Communication in AUTONOTE, A Personal Information Retrieval System", *1971 ACM Conference Proceedings*, pp. 67-81, August 1971.

This article, a brief report on the work reported in full in [Linn 1972], describes a personal retrieval system in which the user can attach simple noun phrases to the items to be retrieved. A semantic network is built from these phrases, and used to maintain a map of what the user is referring to (and to ask questions when the referent is ambiguous). The objective is to make communication with a descriptor indexed text system more like natural language communication between two humans -- in which each builds up a representation of what the other is talking about. Hashcoding techniques are used to access nodes in the network; details of the phrase and node directory structure are given. A technique for maintaining recency of reference information is also used in handling ambiguous referents.

[Linn 1972]

Linn, William E., "Man-Machine Referential Communication in a Personal Information Retrieval System", Ph.D. thesis, University of Michigan. Available from University Microfilms, Ann Arbor, Michigan., July 1972.

This retrieval system, AUTONOTE2, enables a user to define the topical content of each text item with descriptive noun phrases, and to specify structural associations among the defined topics. Internal representations of each topic are formed using syntactic dependency analysis; these are stored in a hierarchical semantic representation network. The net is used to disambiguate queries, and to generate pertinent questions to the user about the meaning of a query. The general motivation is to make more terse and efficient the man-machine communication that goes on in organizing and retrieving personally generated textual materials. The dissertation includes detailed descriptions of the design and implementation of AUTONOTE2, and a case study of system performance during the description of a realistically diverse document collection. Protocol analysis techniques were used. The results suggest that such referential mechanisms constitute a viable alternative to keyword indexing for personal retrieval systems.

[Loring 1971]

Loring, R. J., "Guidelines for a Personal Information-Retrieval System", *Chemical Engineering* Vol. 78 No. 16 pp. 91-95, July 1971.

This paper suggests that engineers may still need a personal system even if they have access to a library that features automated retrieval. It describes how to set up a personal manual document filing system. It is recommended that the user of the system not delegate the selection and filing of information to someone else. The possibility of using commercially available retrieval systems as sources of data is mentioned.

[MacKenzie 1977]

MacKenzie, H., and Kelly, G., "A Query/Update Package for Library or Personal Use", *Australian Computer Journal*, Vol. 9 No. 4, pp. 155-158, November, 1977.

This article describes the implementation, in FORDATA - a FORTRAN dialect which implements the CODASYL DBTG proposals - of a package for accessing bibliographic databases. A high level interactive language is provided for updating and set-oriented querying. Examples are given with a database of about 1000 library books, illustrating boolean queries on author, title, or keyword, and unions and intersections of previously formed sets. Enough information on database structure and implementation is given to augment or adapt the system to other library or personal applications.

[McEowen 1969]

McEowen, James R., "PAL: A Design for a Personal Automated Library", Ph.D. thesis, Moore School, University of Pennsylvania, available from University Microfilms (order number 69-21398), 1969.

This dissertation provides a design for a self-contained personal automated library, to be used for "organization, storage, and retrieval of papers, books, articles, reports, notes, memoranda, letters, and the like". The system allows user definition of multiple record types, building of indices, use of controlled or uncontrolled indexing vocabularies, and even a form of automatic indexing from text fields. An interactive method for making a hierarchical classification of the records (on the basis of descriptors) is explained in some detail. Associative trials can be added to link items together. The file structure is basically sequential, with inverted files used for indices. Some attention has been paid to human factors (a 'digress' button that stacks working context, on-line help, use of menus...). No implementation was attempted, but the design is quite complete.

[Mittman 1975]

Mittman, Benjamin, and Borman, Lorraine, *Personalized Data Base Systems*, Melville Publishing Company, Los Angeles, 1975.

This book is about personalized or customized applications of the RIQS retrieval system developed at Northwestern University. It begins with an overview of bibliographic retrieval and database management systems, and goes on to describe in some detail the RIQS system. RIQS is a conventional 'file of numbered records' system, allowing alphanumeric, numeric, and date fields and repeating groups. An unusually rich on-line query language allows boolean queries with the usual relational operators and "CONTAINS" for text searches. Search results may be saved in 'request sets', which may be intersected, complemented, etc. Computational facilities (including a link to a statistical package) are present, as are plotting commands. Bibliographic applications

have available a variety of printed index production facilities: KWIC and KWOC indices, and sorted author and subject listings. The query language also contains primitive IF and FOR commands. The bulk of the book consists of detailed case studies of how RIQS was used with various research databases in the social sciences, medicine, linguistics, university administration, etc. These descriptions, written by the users, include good descriptions of the problem areas, the database definition used, discussions of any problems encountered in building the RIQS database, and many illustrations of usage of the database.

[Morrill 1978]

Morrill, Lyall Jr., "WHATSIT Reference Manual", Available from Information Unlimited, 331 W. 75th Place, Merrillville, Indiana 46410, 1978.

This easy to read manual describes a microcomputer-based 'database manager' with an engaging, chatty style of interaction. Using a very natural command syntax, the user enters object-attribute-value triples of information (e.g. "BOB'S PHONES'S 234-4567"), and queries ("WHAT'S BOB'S PHONE?") or edit commands. Written in North Star disc basic, the system runs on any 8080 class microcomputer with an S-100 bus, floppy discs, and at least 24K bytes of RAM. Program listings and operating instructions are included in the manual. Triple elements are limited to 30 characters in length, and database capacity is 1500 to 3000 entries per disc, depending on average entry length. A new version, utilizing the CP/M operating system, will store up to 10,000 entries. Response time to commands is at most a few seconds. The package is described as suitable for home use (e.g. indexing your record collection) as well as many small business applications (appointment schedules, customer records,...).

[Nelson 1965]

Nelson, Theodor H., "A File Structure for the Complex, the Changing, and the Indeterminate", *Proceedings of the ACM 20th National Conference*, pp. 84-100, 1965.

This is an early paper on the "hypertext" idea - speculations on what kinds of text structures we need if we are to use computers for personal databases and as an adjunct to creativity (e.g. for manuscripts in progress). 'Hyper' here connotes extension and generality (as in "hyperspace"). Hypertext is a collection of objects so connected and linked that they cannot be composed sensibly into a linear medium. The author reviews the MEMEX idea, then describes the text management problems of the creative writer (emphasizing rearrangement and reprocessing). He goes on to detail the design of an Evolutionary List File (ELF), made up of ordered lists of text items augmented by connecting links.

[Nelson 1967]

Nelson, Theodor H., "Getting It Out of Our System", pp. 191-210 in *Information Retrieval: A Critical View*, George Schechter (ed.), Thompson Book Co., 1967.

This early and interesting paper introduces the notion of non-linear text -- along with the idea that effective personal information retrieval may require more than just document or fact retrieval. It may, instead, be best supported by networks of text fragments, ideas, notes, etc. which are most easily explored via good CRT display facilities. The author calls such a network "hypertext". The model application underlying his thinking seems to be that of an on-line "state-of-the-art review paper", one which is constantly updated and re-organized as work progresses.

[Norton 1970]

Norton, John H., "Setting Up a Personal Information Retrieval System", *Management Review*, March 1970, 1970.

While this article describes a manual system ("without computers, and without any computer programs"), it does contain a good description of the personal information retrieval problem -- especially as it applies to managers. The system described is of the 'peek-a-boo' type, using a keypunch card for each document. The user keeps manual indices of keywords used (and their corresponding numbers), and of documents and their file numbers. A keypunch card is prepared for each keyword, with a punch for each document that is indexed under that keyword. Querying is done by superimposing the cards for the keywords of interest, and seeing if light shines through any holes. The author notes that the system is limited to small document collections.

[Purdy 1971]

Purdy, J. Gerry, "ACCESS - A Program for the Catalog and Access of Information", Report STAN-CS-71-210, Computer Science Dept, Stanford University, March 1971.

This report describes a batch oriented system intended for handling personal libraries. It runs on an IBM 360, and is written in FORTRAN. Produces a KWIC listing of titles, listing by location codes, and listing of titles by user-assigned category codes.

[Reitman 1969]

Reitman, Walter R.; Roberts, R. Bruce; Sauvain, Richard W.; Wheeler, Daniel D; and Linn, William E., "AUTONOTE: A Personal Information Storage and Retrieval System", *Proc. 24th Nat. Conf. Assoc. Computing Machinery*, p.67-76, 1969.

This paper describes a personal storage and retrieval system providing mechanisms for organizing and describing textual information. The user enters unformatted text items, which may include a variety of textual materials, and then assigns descriptors and phrases by which these materials may be retrieved. Typical materials entered include notes, bibliographic references, drafts of papers, ideas for further research, transcripts of discussions, etc.. These text items are roughly paragraph size, but are otherwise unformatted. The usual forms of boolean keyword retrieval are available. An interesting facility is that of a composite descriptor: a user defined string which the system expands into a boolean keyword expression. The user has mechanisms for deleting, replacing, linking, and hierarchically organizing text items. The system is implemented in assembly language for the IBM 360/67, under the MTS operating system, and is accessed via a 120 CPS display terminal. One of its uses is to aid in a study of scientific thinking - protocol taking machinery is built in to aid in such applications.

[Reynolds 1976]

Reynolds, C. F., "A Data Base System for the Individual Research Worker", *Proceedings of the International Symposium on Technology for Selective Dissemination of Information*, 1976.

This paper reports on an unconventional retrieval and text processing language called CODIL (COntext Dependent Information Language) implemented on an ICL 1903A computer at Brunel University in England. The intent is to let individuals define and use their own databases. In CODIL, both data and commands are entered in an indented attribute-value format. Facilities are present for retrieval, sorting, editing, reporting, and reading in fixed format data from other sources.

[Robinson 1973]

Robinson, M. G. and Yates, D. M., "The SCRAPBOOK Information System", *The Information Scientist*, pp. 135-143, December 1973.

SCRAPBOOK is a CRT accessed system supporting text preparation: "...building, altering, and consulting organized bodies of data. Users were encouraged to stick all kinds of data in it - hence the name SCRAPBOOK. The data-base organization, user facilities and security aspects of this system are discussed. It is designed for fast, interactive use. Records (made up of paragraph-sized "frames") may be retrieved by name. There is a hierarchical directory of record names. The sequence of frames displayed is stored, and there are commands to go back in this sequence. SCRAPBOOK has been in experimental use at the National Physical Laboratory in England for some time; some examples of applications are given.

[Sargent 1978]

Sargent, George F.; Droegemueller, Lee; Bell, Norman T., "IRIS, An APL Based Interactive Relational Information System", *NCC '78 Personal Computing Digest*, pp. 235-243, June 1978.

This paper describes a relational database system for the IBM 5100 personal-sized computer. It is capable of working with two-dimensional flat files of the user's own design. Data entry, modification, boolean searching, and reporting facilities are all present, and are illustrated with examples. The user interface relies heavily on menus, and on 'latent prompting' (empty response to a request for input causes listing of the available choices). The possibility of converting to other microcomputers (as APL becomes more available) is mentioned.

[Saunders 1977]

Saunders, Robert J. D., "An On-line Computer Assisted information Retrieval System Using a Minicomputer (CAIRS)", *Program*, Vol. 11, No. 1, pp. 16-30, January 1977.

This paper describes a mini-computer system used by a group of three people to maintain and search a collection of about 30,000 documents in a technical library collection at Leatherhead Food Research Association in England. A Texas Instruments 980 computer, with disks and a 240 CPS display is used. Software, done mostly by an outside software supplier, is in assembly language, and took several person-years to design and implement. A subject keyword index is maintained, and searching is by keyword or serial search of a field (possible limited by an accession number range). Quite a full range of services are present - data entry and editing, searching, hardcopy

output of various forms, thesaurus maintenance, and selective dissemination of information.

[Sauvain 1970]

Sauvain, Richard W., "Structural Communication in a Personal Information Storage and Retrieval System", Ph.D. thesis, University of Michigan. Available from University Microfilms, Ann Arbor, Michigan, Order No. 70-21782, 1970.

This study involved analysis of actual usage patterns of the AUTONOTE system used with a personally generated descriptor-indexed textual database. Detailed protocol records of regular use of the system over a 10 month period were kept. These were analyzed for man/machine communication bottlenecks in imposition of retrieval-facilitating structure. Common difficulties occurred in four areas: keeping track of prior usage of descriptors; translating the user's internal conceptual structures to database structures; determining the structural context of retrieved data items; and minimizing the short term memory load of the user. Techniques for overcoming these difficulties were proposed, primarily as improvements in the system's command language.

[Smith 1969]

Smith, J.E., Butler, J., and Grosvenor, D., "KEYWORD-INDEX. A Computer Oriented Personal Reference Retrieval System.", *Methods of Information in Medicine*, Vol. 8, No. 3, pp.152-154, 1969.

The system reported on is an index preparation program operating from punched cards. Keywords are indicated on the input cards by preceding them with an asterisk.

[Sterner 1976]

Sterner, Ray T., and Breidenstein, Charles P., "Set of ADVANCED-BASIC Programs for Literature Storage and Retrieval with Minicomputers", *Behavior Research Methods & Instrumentation*, Vol. 8 No. 4, pp. 397-398, August 1976.

The authors describe a system (implemented in BASIC) for retrieving literature references by boolean searches. Three digit numeric keyword codes are used for search keys. The software is claimed to be adaptable to a variety of mini-computers, and listings are available without charge. The particular implementation described in this paper uses a Wang 2200B computer, with 3 floppy disks and a 30 CPS printer.

[Szonyi 1974]

Szonyi, Geza, "DRS - A User Oriented Information Retrieval System", *Journal of Chemical Documentation*, Vol. 14, No. 2, 1974.

Motivated by a desire for a FAMULUS-like system, but with random access searching and implementable on a mini-computer, the author used a commercially available data retrieval system, DRS, distributed by Aeronautical Research Associates of Princeton. This system was installed on an IBM 1130 and on a General Automation 18/30, and is in use for library journal routing, laboratory data reports, individual collections of reprints, etc. The article shows as an example a chemical literature retrieval system using Wiswesser Line Notation. Very detailed information is given on database setup, operating procedures, card formats, etc.

[Tharp 1978]

Tharp, Alan L., "Augmented Transition Networks as a Design Tool for Personalized Database Systems", *ACM SIGIR FORUM*, Vol. 13 No. 1 (Proceedings of the First SIGIR Conference on Information Storage and Retrieval), pp. 2-13, May 1978.

This paper discusses application of a transition network technique to implementation of command parsers for personal retrieval systems ("those too large to do manually, but too small to interest a commercial vendor"). The augmentation referred to in the title is to allow tests on arcs of the transition network. An implementation (in SPITBOL) of a small bibliographic search system, using the augmented transition network (ATN) approach, is briefly described. Among the advantages claimed are: ease of implementation; naturalness of the ATN representation of the man-machine interface; and easy generation of informative error messages.

[Van Ree 1976]

Van Ree, T., "A Personal Reference Retrieval System", *Journal of Chemical Information and Computer Sciences*, Vol. 16, No. 3, pp. 152-153, 1976.

This article describes a literature reference retrieval system implemented on a programmable calculator, with tape cassette storage of data, and a thermal printer for output. In indexing, keywords are reduced to 4 byte codes. A cassette tape can contain about 900 references, enough for "about two years normal accumulation of references".

[Walker 1967]

Walker, D. ., "SAFARI, An On-Line Text Processing System", *Proc. American Documentation Institute Annual Meeting*, Vol. 4, pp. 144-147, 1967.

SAFARI was an experimental system implemented on an IBM 7030 computer. It was designed to assist information analysts in building their own personal files. Content representations for news items and the like are constructed from 'kernels' - simple declarative phrases picked with lightpen from text displayed on screen. Context-free parsing is performed, with the computer asking for part-of-speech classification help if it needs it. A content representation made up primarily of subject, object, place, and time of the activity is stored. Retrieval is by light-penning successive menus formed from the content representations of the documents. Editing, and cut-and-paste facilities are present. Some ideas for further research are presented.

[Wallace 1966]

Wallace, Everett M., "User Requirements, Personal Indexes, and Computer Support", *American Documentation Institute, Proceedings of the Annual Meeting*, Vol. 3, p. 73-80, 1966.

This paper is about an early system called SURF for production of personalized printed indexes (personalized in the sense that the indexing is done by the user) . SURF was implemented with the SDC-developed MADAM programming language on the IBM 1401. The article points out that perusal of personal indexes can be useful to central library services in providing better information flow to users. It also mentions the "users become better indexers" phenomenon " ... users have found that they learned to build better indexes through having to use the products of their earlier indexing decisions". Also discussed is the need for maintaining personal files, and the inappropriateness of

'general purpose indexing' for same.

[Warren 1977]

Warren, Jim, "Personal and Hobby Computing: An Overview", *Computer*, Vol. 10 No. 3, pp. 10-22, March 1977.

This is a good hardware/software survey of the hobby and personal computer scene. The author believes that personal computers will have a strong impact, though software, and to some extent hardware, is currently at a relatively crude level. For example, few machines have even file-oriented operating systems. He notes growing interest in word-processing applications, and that relatively primitive database processors are beginning to appear. He predicts that softcopy libraries, key-worded electronic want-ads, and community bulletin boards will be among the applications we will be seeing 5 to 10 years from now.

[Yerke 1969]

Yerke, Theodor B. et. al., "FAMULUS: A Personal Documentation System... User's Manual", U.S. Forest Service Report - available from NTIS as PB-202 534, 1969.

This manual contains operating instructions for all 8 modules of the FAMULUS system, operating on the CDC 6400 & 6600, IBM 360/40+, and UNIVAC 1108.

[Yerke 1970]

Yerke, Theodor B., "Computer Support of the Researcher's Own Documentation", *Datamation*, Vol. 16 No. 2, pp. 75-77, February 1970.

This is a good introductory article to the FAMULUS system. FAMULUS is intended to provide users with an easy mechanism for creating their own files, using their own style, and organized for their own applications. The user interface is designed to be simple enough for non-sophisticated users. FAMULUS is viewed as an alternative to centralized library services; there is some discussion of the relationship between library services and personal databases.

XEROX

PALO ALTO RESEARCH CENTER
ADVANCED DEVELOPMENT LABORATORY

Date August 4, 1978

To: Whole Alto World

From: David Cronshaw

Subject: Miscellaneous ALTO "bits and pieces"

Over the past year (or longer) I am sure that a significant amount of hardware, micro-code and software has been developed by the vast numbers of people who have had occasion to interface an ALTO to some alien piece of hardware for the sake of their pet project. Whilst some of this hardware etc. has been documented formally and "released" to the Whole ALTO World, I am sure that much of it lies buried in closets finding little interest amongst other than those in its immediate vicinity. I therefore propose that the WAW group and its newsletter could serve as a vehicle for publicising the existence of such items in the hope that some reduction of duplicated effort might be achieved.

To start the ball rolling, I would like to describe a couple of "tidbits" which have been designed and built at PARC/ADL and which have potential for use on other projects involving ALTOs.

1. A Buffered Interface for ALTO

Designed and in use as an input-output buffer for use with a CCD scanner and a TRIDENT disk. Data is buffered in 2 k x 16 RAM buffers. Operation is half-duplex in either direction, and data rates of up to 10Mhz can be supported into memory or up to 6Mhz to or from the TRIDENT. Figure 1 shows a block diagram of the logic which interfaces with the ALTO processor bus. The data connection to the CCD is serial, though 16 bit parallel data could easily be provided for some other device. Technology is standard TTL and static RAM on a wire wrapped AUGAT board plugged into the ALTO backpanel.

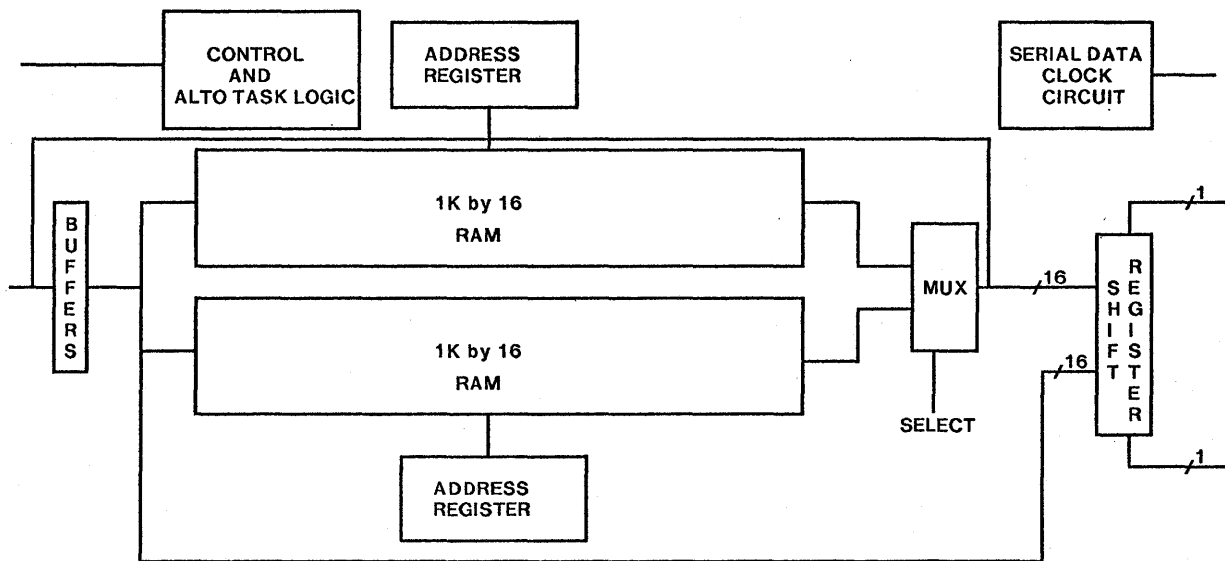


Figure 1. Buffered ALTO Interface

2. A High Speed Serial Data Interface for ALTO?

Primarily designed as a parallel to serial/serial to parallel converter module for use in an optical fiber communications experiment being conducted at PARC/OSL, a module has been designed and built which takes 16 bit words in at one end and throws out a Manchester encoded bit serial stream at up to 150Mbits/sec and generates a 16 bit CRC at the same time. In the reverse direction, Manchester encoded serial data is decoded, converted to 16 bit parallel form and CRC checked for error detection. A block diagram of the module is shown in Figure 2. The technology used is high speed ECL (Fairchild F100K) and multi-layer board. Power supplies required are -5.2v, -2v and +5v. Space is available on the board to add ECL-TTL level changers if required.

3. A High Speed Buffered Interface for ALTO?

By combining the above two modules, a buffered high speed interface could easily be built for an ALTO providing for instantaneous data rates at up to 150Mhz for packets of data up to 16k bits long.

If anyone is interested in any of the above, or in combination, they should contact either myself (Dave Cronshaw, IntelNet 823 7279) or Warren Sterling (IntelNet 823 7126)

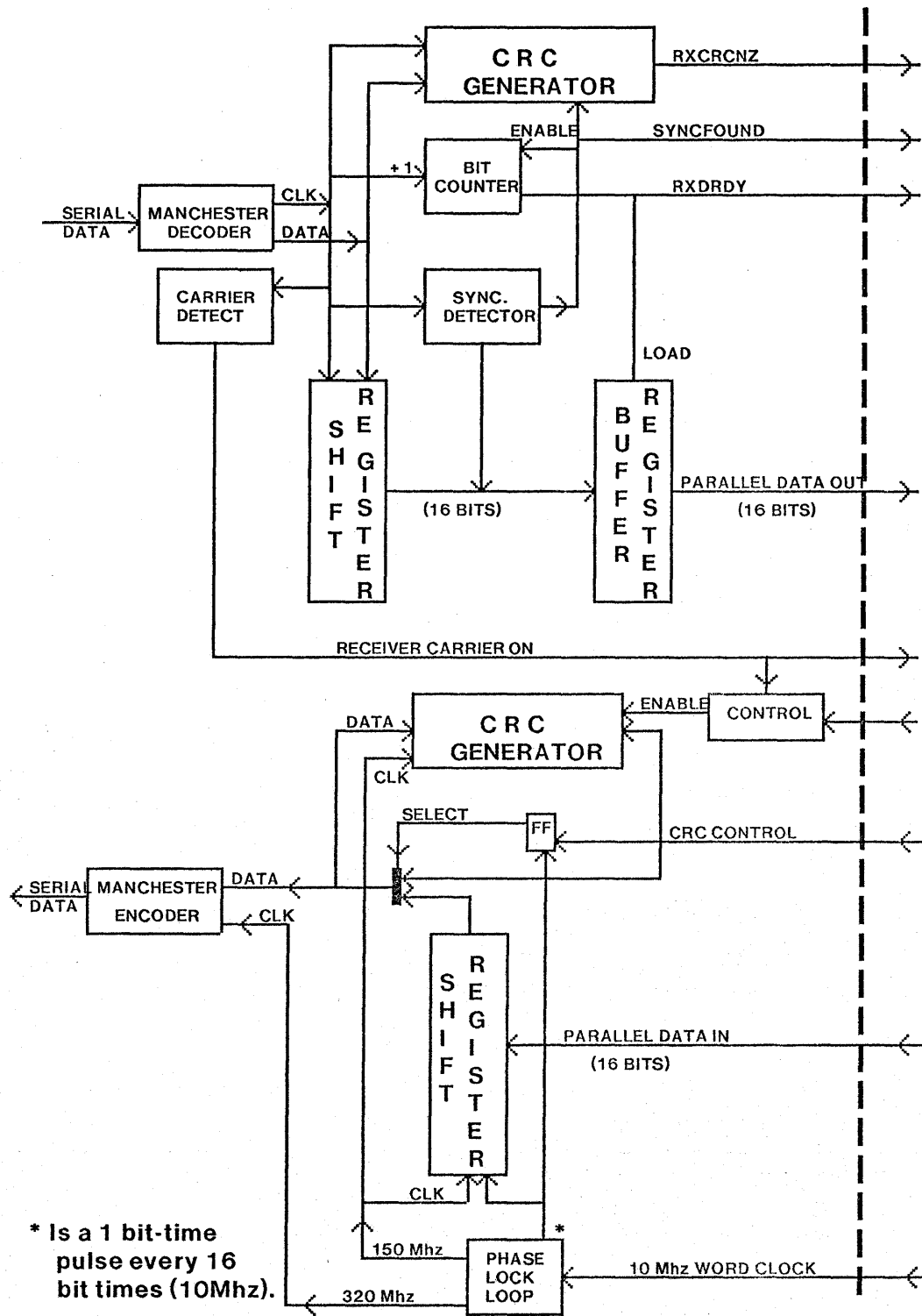


Figure 2 High Speed Serial to Parallel Converter

Whole ALTO World Newsletter

Technology and Tools

XEROX

October 15, 1978

SPECIAL ANNOUNCEMENTS

WHOLE ALTO WORLD MEETING - The Whole Alto World meeting is being jointly hosted by Jim Iverson of WRC and Darwin Newton of Ginn this Tuesday, October 17th, at the Le Champignon Restaurant in Webster, New York. Discussion topics include the Alto Network and Alto Gateways, recent hardware and software developments, and activities at various user sites. A report on the meeting will appear in the next Newsletter.

GENERAL NOTES

NEW ALTO NETWORK COORDINATOR - As previous Newsletter items have indicated, WRC/RTCC is assuming Alto Network operational responsibility from PARC. Art Axelrod of WRC is the new coordinator. Art will coordinate the installation and operation of Gateway systems, phone lines, etc., and maintain the name lookup data base. Inquiries about setting up new gateways and requests to add or change machine names should be directed to Art at 8*222-5811 or message <Axelrod>. Many thanks to all those at PARC that handled the network operational details in addition to their regular jobs.

USING THE TERM "ETHERNET" - A memo was recently received from Barry Smith of OGC/Patents concerning the use of the term "Ethernet". The text of the memo follows:

As you may know, we eventually will be using the term "Ethernet" as a trademark and applying for trademark registration. Accordingly, it now becomes important for us to insure that term does not become generic, which would prevent us from obtaining federal registration.

Accordingly, please insure that future Xerox publications use the term "Ethernet" only as a adjective to modify the generic system, e.g., and Ethernet multi-access communications network. The term Ethernet must not be used as a noun, e.g., an Ethernet.

MARKET PLACE

Market Place provides a forum for Alto users to make offerings and requests for Alto related hardware and software. To place an "ad" or offering, send the text to the coordinator, Frank Ludolph, at PARC, message <Ludolph>, or phone Intelnet 8*923-4652.

BREAKING UP LARGE FILES - Geoff Thompson offers a tool provided to him by Bill Duvall to break a large file into smaller ones. The specific use in this case was for crash recovery. The program, Breakfile, was used to break a large Garbage.\$ file into smaller pieces for editing with the Bravo unformatted Get. The program is on [IVY]<Thompson>Breakfile.run

Whole ALTO World Newsletter

A STANDARD P-BUS INTERFACE - Geoff also offers a standardized P-bus interface for the Alto. SIL drawings for the interface are available from [IVY]<Thompson>StanInt01.sil, StanInt02.sil, ... StanInt05.sil, and StanIntBdMapA.sil.

SOME MICROCODE STUFF - Dave Cronshaw has written some microcode routines for extended memory versions of the Convert, Block Move, and Block Store instructions, and has written some BCPL routines to make use of the ASIM package easier for checking out new emulator task microcode. Formal documentation does not exist; it will be written if interest exists. Contact Dave at 8*823-7279 or message <Cronshaw>.

TOOLS

HARDWARE

8TH BUILD - Shipments of the 8th build have begun ahead of schedule. Though SPG cannot commit to an accelerated shipping schedule due to a less than normal inventory of modules, they will attempt to maintain the current lead to the extent possible. On a personal note, thanks to all the SPG people for their efforts and the hassles the early shipments saved us in ASD User Support Services.

ALTO I POWER UP HAZZARD - Simply powering up an Alto I and setting the disk to RUN can potentially clobber the disk. A fix is being investigated but in the meanwhile the proper sequence of actions is: 1) power up with the disk in LOAD, 2) push the boot button, 3) set the disk to RUN.

MAINTENANCE NOTES

ALTO XM PHANTOM PARITY ERRORS - Extended memory Altos are reporting parity errors in the second, third, and fourth memory banks which diagnostics fail to replicate. The reported errors, in fact, do not exist. DMT sets parity indicators during its normal operation. The Operating System, when booted, resets these indicators for the first bank (64K) only. The phantom errors, which occur only with programs that use more than 64K, are a result of uncleared indicators in the higher memory banks. The next release of the operating system will reset all banks; in the meanwhile extended memory programs should clear them during initialization.

If you are running an extended memory program which fails to reset the parity indicators, Bill Duvall offers a small program which will reset them; simply execute it immediately after booting or prior to running the XM program. Retrieve [IVY]<Duvall>FixX.run.

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local IVY server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [XEOS].

Whole ALTO World Newsletter

NEW RELEASE: CALLUP - This experimental program from Dan Swinehart will lookup the telephone number of an individual on an on-line directory and, if a "Ross box" is attached to your machine, place the call. The directory is taken from the Xerox Intelnet directory and can be extended by personal listings if you desire. The program is fully documented on [MAXC]<Swinehart>Callup.press.

ReReleases - Subsystems

BINLIST - This program, used with PROM and PNEW, now has their switches, e.g. P, Q, N, etc. Retrieve BINLIST.run.

LAUREL - Version 2 is based on Mesa 4.1 and incorporates several new features including a functional Hardcopy command and new Get and Put commands. Installation is somewhat different so it is suggested that you retrieve and read the revised documentation on [MAXC]<Laurel>LaurelManual.press.

PREPRESS - Version 1.9 incorporates the Menu package into the "Edit" function and fixes a n Impose Widths bug and now enables the user to call ImposeWidths with arbitrary source and destination files. Retrieve PREPRESS.run. Documentation on the new features is on PrePress-News.bravo.

PRESS - The latest version contains several bug fixes. There are no changes in the installation procedure. Retrieve <Press>PRESS.run.

PROM/PNEW - These two programs now accept MB files with fixups and fixes some bugs. Retrieve PROM.run or PNEW.run as appropriate. The documentation, PromManual.press has been updated.

PUPTEST - The nature of the changes are unknown to me. Retrieve PUPTEST.run.

ReReleases - Packages

ASIM - This package has been enhanced to simulate all the memory reference capabilities of Alto IIs and of Alto Is with the doubleword store modification. Retrieve ASIM.br and ASIM.d. Revised documentation is available on ASIM.tty.

TECHNOLOGY

This month's paper by Wayne Wilner presents a functional description of a novel information-processing architecture claimed to be well suited to office needs. It stretches the imagination by offering an alternative to the von Neuman machine.

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC]<AltoDocs>WawNewsM-YY.press or may be obtained from the editor, Frank Ludolph, ASD, by messaging <Ludolph> or calling Intelnet 8*923-4652.

XEROX

PALO ALTO RESEARCH CENTER

Systems Science Laboratory

LSI Systems Area

January 21, 1978

To: Association for Recursive Machinery

From: Wayne Wilner

Subject: Recursive Machines

Stored: IVY <Wilner> RecursiveMachines.press

This memo contains a functional description of a novel information-processing system architecture, called "Recursive Machines", which is proposed as a viable alternate to von Neumann architecture and is claimed to be well suited to office information-processing. "Functional description" implies that neither implementation nor use nor evaluation is within the memo's scope; these must be treated in sequel.

0. MOTIVATION

This work is motivated by the greatness of the disparity between our contemporary models of office information-processing and the traditional model of information-processing systems, the von Neumann computer. Information in an office is represented by a growing, adaptive set of flexible structures; information in a von Neumann computer is represented by bytes in a rigidly bounded, linear address space. Office procedures are a growing, adaptive set of loosely interconnected, event-driven activities; procedures in a von Neumann computer are a rigid, control-driven sequence drawn from a small set of logical operations. Is there a viable, alternate model of information-processing systems which is similar to, rather than disparate from, our models of the office?

1. ARCHITECTURAL PRINCIPLES

1.1 Basic Assumptions

Our most basic assumption is that it is time to use a richer model of information than a 16-bit word. Existing storage structures are so simple that they have come to be one of the disadvantages of traditional architecture, instead of one of the advantages. They do not model real-world data well, and the cost of overcoming their structural simplicity is greater than the cost of using a richer

set of storage structures. This is true without question at the software level; we assert it has recently become true at the hardware level. Recursive Machines handle data cost-effectively by using an hierarchical, variable-length structure as the most basic element of storage.

Our second basic assumption is that it is time to use a richer model of processing than a sequential machine. Existing processors are so simple that they have come to be one of the disadvantages of traditional architecture, instead of one of the advantages. Having a single, centralized, sequential, register manipulator discourages the use of two vast classes of algorithms: descriptive rather than procedural, and concurrent rather than sequential. Recursive Machines encourage the use of a wide class of algorithms by having an unlimited number of generalized sequential machines as basic processing elements and by controlling them in a decentralized, concurrent manner.

Our third assumption is that as LSI circuits shrink and speed up, the limiting resource in an information-processing system will no longer be main store nor processor cycles but *pins*. Indeed, some observers define VLSI to be that level of circuit technology for which traditional architecture becomes less advantageous than some alternate architecture. Recursive Machines use three tactics for coping with pin limitations: (1) perform operations serially to minimize widths of communication paths, (2) locate data, program, and interpreter together to minimize off-chip communication, and (3) decentralize systems functions, such as scheduling, storage management, and input/output operations to reduce system-wide communication.

Our fourth assumption is that there will be further miniaturization of components by three orders of magnitude, making physical boundaries coincide with different interfaces during a machine's lifetime and making component failures certain. Recursive Machines cope with increasing miniaturization by: (1) making any collection of elements functionally identical to a single element so that chips with 10 and 1,000 elements are functionally identical, (2) using logical, rather than physical, addresses, (3) being basically asynchronous, and (4) integrating recovery mechanisms.

1.2 Basic Principles

The purpose of a Recursive Machine is that of an information-processing system: to hold information and to allow that information to be examined and changed. VLSI gives us both the necessity and the opportunity to create wholly new information-processing systems. Such latitude requires us to have a set of governing principles which compel all design decisions to fulfill a single purpose. The following principles are offered as being more harmonious with each other and with VLSI technology than with traditional principles.

1.2.1 Recursive storage Traditional storage structure is characterizable as a disjoint collection of media, where a medium is a linear vector of fixed-length cells. For example, {ROM, registers, cache, main store, IO buffers, device storage of various kinds}. Recursive storage generalizes the notion of a storage medium to a recursively-defined hierarchy of variable-length cells. This is sufficiently general to obviate the notion of a collection of media.

1.2.2 Recursive machine organization Traditional machine organization involves specialized components. Storage subsystems, processors, input/output controllers each perform disjoint sets of storage, processing, and IO functions, respectively. Recursive machine organization generalizes the notion of a component to an all-purpose element, capable of storing, processing, and transmitting information, both to other elements and to external devices. Over the range from nand gates to Cray-1, Recursive Machine elements are medium-small. Recursively, any group of elements is all-purpose.

1.2.3 Recursive system organization Traditional systems are implemented by rigid physical structures: components sit on conductors etched in printed-circuit boards, boards are bussed together, racks contain a fixed number of boards. The logical structure mirrors this rigidity. Recursive machine organization generalizes the notion of a component to a recursively-structured, n -connective element. Any number of elements have, as a

group, structural and logical properties identical to those of a single element.

1.2.4 Recursive machine language Traditional machine language involves a fixed set of simple operations on simple operands. Recursive machine language generalizes a machine operation to be either a hard-wired primitive or any recursively-definable composition of machine operations. This allows arbitrarily complex operations on arbitrarily complex operands.

1.2.5 Decentralized, concurrent control In traditional systems, an instruction becomes executable when a centralized, sequential processor fetches it as its next instruction. In Recursive Machines, an instruction is continuously executable. It may receive and transmit data at any time. Its execution may produce outputs which are sent to other instructions, possibly causing them to produce outputs. It is the decentralized flow of data from one instruction to another which controls the system's operation, allowing many instructions to execute concurrently.

The purpose of this memo is to elaborate this statement of principles sufficiently well to allow their implementation.

2. RECURSIVE STORAGE

2.1 Information

We define information in terms of a representation for it. Information is represented, in part, by a delimited string of data characters. Delimited strings directly reflect that property of information that allows words to have a different number of characters in them.

```
unit of storage    ::= "(" character* ")"
character         ::= "0"
character         ::= "1"
```

Example: 2001 = (1111010001) if the most-significant bit is left-most,
 = (10001011111) if the least-significant bit is left-most.

To keep things simple, we choose binary digits for data characters. To allow strings to be as long as possible, Recursive Machines control operations on items by counting matching delimiters, not by counting characters. This means that in principle there is no limit to string length other than total storage capacity.

Composite and nested information can be represented by letting an item of storage recursively be a delimited set of items of storage.

```
item of storage   ::= "(" item of storage* ")"
item of storage   ::= unit of storage
```

Example: "An item may have either unordered or ordered subitems." =

Sentence		
Subject	Verb	Object
Article Noun	Auxiliary Infinitive	Prepositional phrase
(((An) (item)) ((may) (have)) ((either unordered or ordered subitems)))		

This definition represents everything as a tree-structure. In principle, there is no limit to the number of subitems in an item, nor to the depth of nesting. Notice also that non-circular lists, queues, varying-width stacks, etc., are representable without using pointers or addresses. This does not mean, though, that pointers cannot be used to represent cyclical structures.

An item may have either ordered or unordered subitems; that is, it may be either a set or a sequence. An unordered item may have subitems which are either identical or distinct; that is, it may be either a set or a bag.

2.1.1 Implementation

Although this memo is a functional description of Recursive Machinery, some comments on implementation are necessary to emphasize differences from von Neumann architecture. One implication of recursively defining an item of storage to be an ordered set of items is that it must always be possible to place another item between any two items. A principle advantage of delimiters is that they allow RM storage cells to perform such insertions using a tiny amount of logic. Using delimiters, it is possible to distinguish physical storage which contains items from that which does not. Let us call storage which does not contain an item "unused space". We can distinguish it from used space by redefining an item of storage to be a set of items separated by unused space.

```

item of storage ::= "(" item of storage (space* item of storage)* ")"
space          ::= "0"
space          ::= "1"

```

This definition has three advantageous properties. First, it accounts for physical space without affecting the logical structure of any item. The nth item in a set is always the nth item, no matter how much unused space surrounds it. Second, it accounts for physical space in terms of the logical structure of data, making it easy to do physical allocation as a part of logical storage operations. Third, it requires only two characters to distinguish all structural boundaries. Let "∂" stand for zero or one; then:

```

"...∂(..." is the boundary between unused space and an item;
"...((..." is the boundary between an item and its inferior;
"...(∂..." is the left end of a unit;
"...∂)..." is the right end of a unit;
"...))..." is the boundary between an item and its superior;
"...∂)∂..." is the boundary between an item and unused space;
"...()..." is the empty unit;
"...)(..." is the empty space;

```

reducing the logical complexity of the primitive operations.

2.1.2 Encoding

Our model of information requires four distinguishable characters, so we can use quaternary digits. We can shorten this term to "quats", and that gives us four suggestive names for each digit, viz.:

```

"1"    high-quat
"0"    low-quat
"("    come-quat
")"    go-quat

```

This memo is written in terms of a four-character alphabet, or, more properly, two two-character alphabets, one alphabet for data and another for the shape of the data. This does not prohibit larger alphabets, such as two 256-character alphabets.

The null item is vitally needed as a place-holder in time for our temporal control system, just as zero is needed a place-holder in space for our positional number system. Let us encode null as "()" and zero as "(0)".

2.2 A Warning

Before describing addressing and storage operations, let me caution the reader against a common error. Because of traditional architecture, we expect storage to be one of several distinct subsystems.

This is not true of Recursive Machines. Storage is all, and all is storage. Every Recursive Machine operation is performed on items by items for items. There is no distinction between storage and processing operations: one mechanism does both. There is no distinction between storage addresses and processor addresses: they are in a single address space. There is no distinction between storage and control: they have a single realization.

On the other hand, it helps to grasp the passive and static aspects of the system before attempting the active and dynamic aspects. For this reason, we will explain some specific storage operations before describing how Recursive Machines operate in general. Just remember that items are conceptually both the storage and processing elements of the system.

2.3 Storage Operations

The basic storage operations include "read" and "write" which change the data inside of items, but there must also be operations which create and destroy items themselves. We will call two of these operations "put" and "take". Also, no single orientation for strings is optimal for all common operations; thus, it should be equally convenient to process a string from either end.

2.3.1 Putting

When an item receives a message saying "put" with an item as a parameter, it creates a new item, beside itself, with contents equal to that of the second parameter. There are two messages: "putL" which places the new item on the left; and "putR" which places the new item on the right. [Implementation note: there need not be any unused space beside the item initially. Two adjacent items can always be pushed far enough apart to obtain sufficient unused space. As the items are pushed apart, unused space elsewhere contracts.]

2.3.2 Taking

When an item receives a message saying "take", it sends itself to the sender. It disappears from its former location. There are two messages: "takeLR" which transmits the characters from left to right and "takeRL", from right to left. [Implementation note: if the item is between collateral items, the quats which it occupies may become unused space. Contrarily, if the item has no left-hand or right-hand neighbor, its quats cannot become unused space because this would violate the definition of §2.1.1. "Taking" it must create unused space elsewhere.]

2.3.3 Reading

When an item receives a message saying "read", it sends its contents to the sender. There are two messages: "readLR" which asks for the characters from left to right and "readRL", from right to left.

2.3.4 Writing

When an item receives a message saying "write" with an item as a parameter, it changes its contents to be that of the parameter. The new contents may be shorter or longer than the old [creating or consuming unused space elsewhere]. Also, the new contents may have different structure from the old.

We might portray the various operations as follows.

	Before: <u>...)(item)(...</u>	
Message: (item) (putL) (abcd)		Message: (item) (putR) (abcd)
After: <u>...)(abcd)(item)(...</u>		After: <u>...)(item)(abcd)(...</u>
Message: (item) (takeLR)		Message: (item) (takeRL)
Reply: (item)		Reply: (meti)
After: <u>...)(...</u>		After: <u>...)(...</u>
Message: (item) (readLR)		Message: (item) (readRL)
Reply: (item)		Reply: (meti)
After: <u>...)(item)(...</u>		After: <u>...)(item)(...</u>
Message: (item) (write) (abcd)		
After: <u>...)(abcd)(...</u>		

2.3.5 Discussion

Since "put" and "take" create and destroy items, they perform stack push, stack pop, enqueue, and dequeue directly, rather than by manipulating pointers. Notice that it is unnecessary to reserve space for stacks and queues. A stack of any length is initially just an empty item, "()". In this case, the item which receives the "put" message is actually the empty contents of the empty item. It is immaterial whether the empty contents puts the new item before or after itself; the result is "((new item))". FIFO operation results from sending all "put" messages to whatever item is at one end and all "takes" to the other end. LIFO operation results from sending all "puts" and "takes" to the same end. Double-ended queues receive "puts" and "takes" at both ends.

Because the primitive storage operations are governed by delimiters, they transfer not only items, but vectors, files, etc. Blocks of information can be transferred without resorting to an iterative program or microprogram.

Since a primitive storage operation can involve millions of bits, separate incoming and outgoing lines are needed to allow other operations to take place concurrently. Also, separate lines are needed to allow faults to be signalled during a transmission.

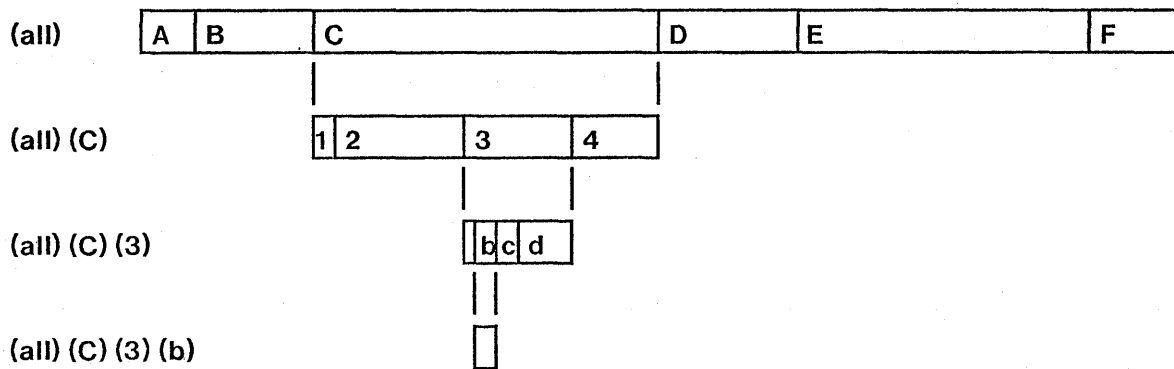
The smallest item, "()", is still long enough to have its come-quat in one physical Recursive Machine element and its go-quat in another physical element. Thus, primitive storage operations require cooperation between elements.

[Implementation note: whether a Recursive Machine element is implemented using RAM, CCD, or disk storage technology makes no functional difference, only a performance difference. Any reasonably large system would use several forms of storage technology.]

2.4 Addressing

Even the simplest operations, such as reading and writing, involve communication between items which can distinguish themselves from other items. It is *distinction* which is the purpose of Recursive Machine addressing, not location. The location of an item changes frequently. [Implementation note: like telephone numbers, RM addresses control switches rather than wires. Also, like mobile phones, items can receive transmissions while moving.]

We can use the recursive composition of information in storage to break addressing into small steps. Consider the general problem of selecting one item out of all. That item is either the root, or one of its inferiors, or is contained in one of its inferiors. Addressing can begin by sending the root a message which selects itself or one of its inferiors. If an inferior is selected, the process may repeat, using it as the root of a subtree. An *address* is a sequence of messages, sent to a sequence of items which sit successively below, beside, or above the previous recipient in the overall tree structure.



Notice that the address of an item is portrayed as a vector of selectors. In order to distinguish an item from all other items, a selector is needed for each superior of the desired item.

It is not usually necessary to distinguish items from all other items. Consider communication between two collateral items (that is, the same superior item contains both as immediate inferiors); they need distinguish themselves only from their superior and any other collateral items.

Distinguishing items based on their position within an overall tree structure or on their content has important advantages: (1) an item's address is independent of changes in its physical position; (2) it is independent of changes in its length or in the length of any item; and (3) it is independent of changes in the allocation of unused space.

Also, there are many changes in the overall structure which leave addresses intact. Examples are: (a) rearranging subitems which are never addressed absolutely; (b) putting and taking subitems from an item which is never addressed absolutely; (c) changing the internal structure of subitems which are always addressed absolutely. Notice that if items are always addressed by sending messages to their superior, then the items' addresses are unaffected by all changes in items not superior to them.

2.4.1 Addresses

Selection can be based on any computable function. Consider the following kinds of address:

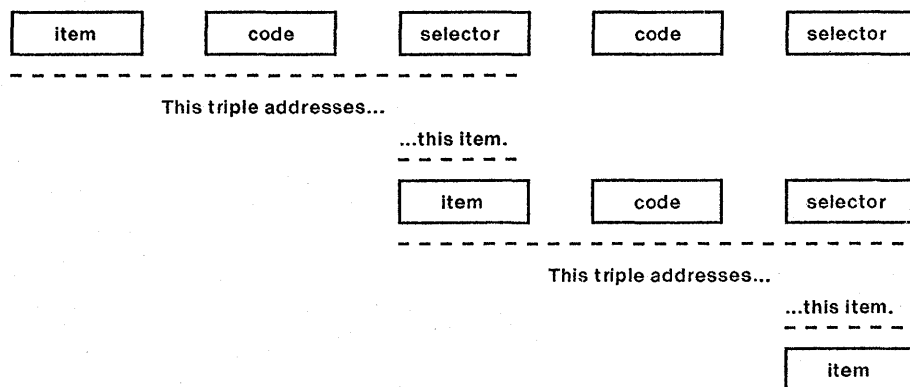
- | | |
|--------------------------|--|
| absolute | subitems are considered to be ordered and numbered (from either end); selection is made by specifying ordinal position; e.g., #1770, #2-#10. |
| position-relative | subitems are addressed by relative, rather than absolute, position; e.g., "last", "every other". |

self-relative	items are addressed in terms of tree-traversing, beginning at a given item; <i>e.g.</i> , "immediate superior", "next".
content-addressable	items are accessed depending on their contents, not on their position; <i>e.g.</i> , "S*", <400.
computed	the item's address is some function; the name of the function appears in the "address" but its value is used as the address (<i>e.g.</i> , successor function, "odd & ~first").

In addition, items can retain information about previous accesses, allowing further kinds of address:

context-relative	the requested item is understood to be in a previously specified subtree; that portion of the absolute address which identifies the subtree is omitted; this could also be called "base-relative".
request-relative	the item is understood to be near the previously addressed item; only the difference between the addresses of the two items is needed; this could also be called "incremental".

Consequently, an *address* is an alternating sequence, in which the odd terms are selectors of these various kinds and the even terms are codes which indicate what type of selector follows. The first term names an item unambiguously, such as "root" or "self". The interpretation of the first three terms of an address, "item-code-selector" yields another item which interprets the remainder of the address.



[Implementation note: in addition to having efficient forms of address, other storage elaborations such as caches, virtual addresses, adaptive organization, etc., may still be useful in different implementations.]

2.4.2 Address space

There is no guarantee that any address is valid, except the address of the root. There must be at least one item in storage in order to perform the operation which creates items.

Unused space has no address associated with it and there is no highest address, so additional elements can simply be plugged in to enlarge the system. This means that trying to stick one more item into a system in which every quatum is physically in use needn't be fatal if more elements are on hand. Contrast this with traditional storage systems.

2.4.3 Logical-to-physical mapping [Implementation note]

A crucial difference between Recursive Machines and traditional machines is that logical addresses

are always used and physical addresses are never used. This is possible because:

- (i) All storage belongs to a single tree structure, which can be thought of as a long vector of items.
- (ii) Every physical machine element contains a portion of this vector in its physical store, which can be thought of as a shift register of items.
- (iii) One of the ways in which all physical machine elements are connected is in a vector. This means that all physical storage is connected as one long shift register.

In other words, the logical organization of all information in storage is identical to the physical organization of the store. There is a total ordering of all items in storage and the same total ordering of all physical elements. Consequently, if each element keeps track of the absolute addresses of the first and last quats in its store, then it can tell whether or not any logically addressed item is within its store. Any absolute address which is presented to an entire RM is resolved to a single element.

2.4.4 Broad distinctions

An address may distinguish more than one item. This simply means that ensuing messages will be received by more than one item. Consider retrieval from a large file or document based on content. The file would spread over many elements. Any specific string could potentially be within any element which contained any portion of the file; so, all such elements must search simultaneously to try to fulfill the address request. The larger a file is, the more elements search. Thus Recursive Machines have the delightful property that content-based accesses take constant time; they are not proportional to the size of the file being searched. Specifically, they average half the time required for an element to circulate its entire store. [Implementation note: as an estimate of what this time is, suppose large files were kept in RM elements which held 2048 quats (4Kb) and circulated at 100 MHz; then, content-based accesses would average 10 microseconds.]

2.5 Space Management [Implementation note]

Recall that there may be unused quats between any two collateral items. This means that a "put" operation may indeed find space available for the new item. Similarly, a "take" operation may convert the disappearing item into empty space. Thus, adjacent items don't necessarily move for "put" and "take" operations. Movement depends on the distribution of unused space.

We know that virtual memory systems operate most efficiently when 15% of their real memory is unoccupied. Recursive Machine elements operate most efficiently, too, when some similar percentage of their physical store is unoccupied. When no local items are transmitting or acting on messages, RM elements trade items with their neighbors for unused space to obtain efficient distribution.

3. RECURSIVE MACHINE LANGUAGE

Control of Recursive Machines involves three concepts. First, the basic agent of control is the generalized sequential machine, or "Gsm", which can be represented by an item of storage. Because there is an explicit representation of control as an item, we can say that every Recursive Machine operation is performed by items on items for items. Second, the set of primitive operations is reversibly extendible so that a Recursive Machine can become totally bound to a particular algorithm, and then unbound. This is possible by implementing the basic hardware in a partly dynamic structure, allowing arbitrary Gsm's to become "hard-wired". Third, among the primitive operations are methods for one Gsm to control other Gsm's. In particular, sequential operation is just one of several ways to organize subordinate Gsm's.

3.1 Generalized Sequential Machines

Generalized sequential machines have state, receive inputs, send outputs and change state according to a transition function [Kain]. Inputs can arrive any number of times before any outputs are sent, and outputs can be sent any number of times before an input arrives. This makes them more powerful than the procedures and coroutines of traditional programming languages, which accept only one set of inputs each time they are called and return at most one output for each set of inputs. Gsm's are also more powerful than the kinds of nodes usually considered for data-driven processing [Davis]. For example, Gsm's can describe functions which: (a) have optional inputs, (b) execute when a threshold number of inputs have arrived, (c) time-out, and (d) operate on several instances of one input that are separated in time.

Recursive Machines are programmed by describing Gsm's which perform a desired computation. This means that Recursive Machine instructions are not encodings of what a processor is to do during one cycle; rather, they are continuously executable encodings of what a Gsm does.

Each physical RM element has a number of *processors* which are available to interpret the Gsm encodings held in storage: to record inputs sent to them, to transmit outputs sent by them, to indicate changes of their state.

Execution can take place whenever any input is sent to a Gsm, causing it to change state. Since the receipt of input is performed by the same mechanism which executes instructions, namely, a processor, it is guaranteed that an instruction which is made executable by that input can be executed immediately.

Of course, executable instructions need not execute immediately. Consider an instruction which is in the midst of execution and requires an input which has not yet arrived. It can issue a request for that input to some executable instruction which is programmed to supply it. Receipt of a request makes a processor available to the second instruction, causing it to yield output on demand. Data bases are an example of Gsm's which are naturally driven by the need for output, that is, retrieval.

What we normally think of as a variable can be viewed as a Gsm. It can receive inputs which tell it to put, take, read, or write. This is the same as object-oriented systems [Kay], in which a variable can add a number to its own value, compare its value to another, send a bitmap which portrays its value, and generally perform any operation. We use the term "item" rather than "object" simply to avoid preconceptions associated with objects.

3.2 Control Representation

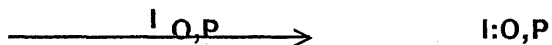
One way of representing Gsm operations is railroad notation, where the machine is thought of as an engine at a particular place on the railroad (representing its state) facing a number of switches, one of which is selected on the basis of the next input and where outputs are produced by traversing a stretch of track which has an expression written on it.

An example for adding two positive integers is given on the next page. This example follows the data-driven notion of addition, in which a "node" expects two addends to be sent on separate input lines and begins to add them when two addends are present as inputs.

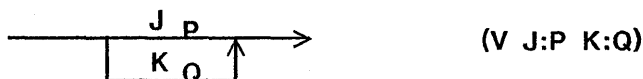
By using Gsm's instead of nodes, other notions are equally feasible. One alternate is to assume the addends are interleaved in time on a single input line. Addition is performed by first absorbing each odd input, then adding the next input to it, then yielding the sum, and finally returning to the initial state.

Basic operations:

receive input I, produce output O followed by output P



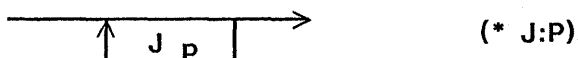
receive input J or K, produce output P if J, Q if K.



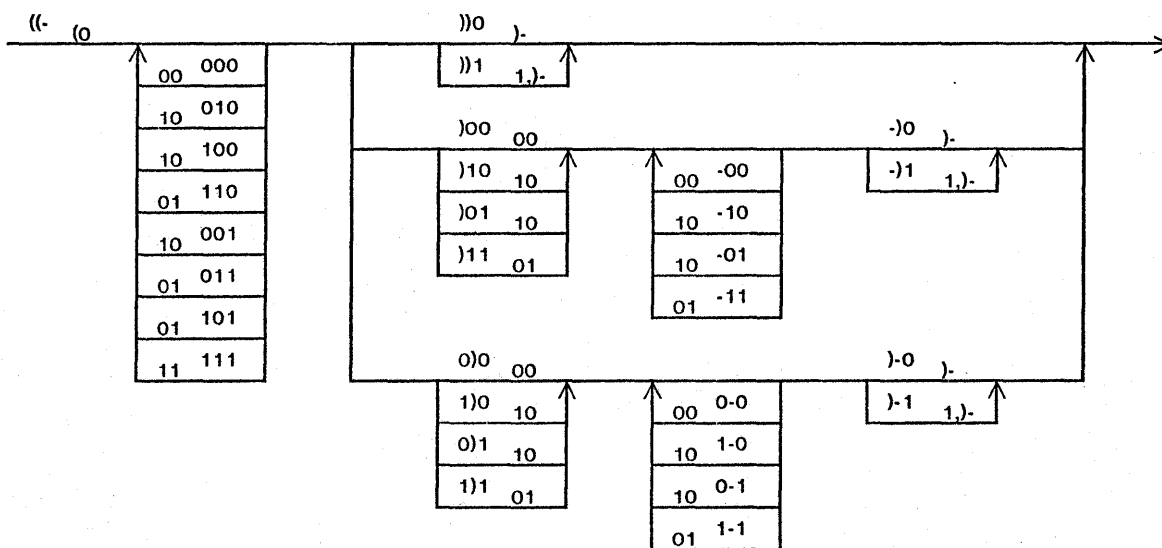
receive input J, produce output P; receive input K, produce output Q.



receive any number of inputs J, produce an output P for each.



Example: to add two inputs which are delimited positive integers, use a machine having three inputs and two outputs, one of them feeding back (for the carry).



Linear representation of the algorithm:

```
(A ((-:(0 (* 000:00 010:10 100:10 110:01 001:10 011:01 101:01 111:11 )
  (V (V ))0:- ))1:1,- )
  (A (V )00:00 )10:10 )01:10 )11:01 ) (* -00:00 -10:10 -01:10 -11:01 ) (V -)0:- -)1:1,- ))
  (A (V 0)0:00 1)0:10 0)1:10 1)1:01 ) (* 0-0:00 1-0:10 0-1:10 1-1:01 ) (V -)0:- -)1:1,- ))
)
```

A machine which is controlled by this expression will, if its two inputs are the strings "(11)" and "(001)", produce the output "(111)"; that is, it will compute 3 + 4 = 7. Two things are implied: the marking which represent the current state of the machine, and control over when to receive inputs and send outputs.

An alternate which builds on this last idea assumes again that there is a single input line. Addends are preceded by an item which describes addition. This is absorbed first, so the addends can be thought of as arriving at a general-purpose machine which has temporarily been conditioned to do only addition. If we think of the input line as the connection from a conventional memory to a conventional processor, we see how close this is to the familiar notion of how instructions execute.

3.3 Extendible Control

Railroad notation can be mechanically converted to And-Or logic. If we imagine building Recursive Machine elements which contain dynamically programmable logic arrays, then we have achieved a system in which the set of "hard-wired" functions is alterable. The set of machine primitives can dynamically include any function. The result is a system from which the machine language level has been removed: all software renders itself into logic functions which are literally hard-wired when they execute.

If we take one more step, namely, control an element either by its own PLA or by a comparable bit pattern taken from storage, then we remove the usual space restriction. This is the same as having a microprogrammable machine execute microinstructions directly from main storage.

Of course, in And-Or form, functions require considerably more storage than in a machine language encoding. For many functions, the machine language level is preferable. In fact, additional levels of encoding are worthwhile. The point is that, no matter how many levels are implemented, if any function can be moved to an adjacent level mechanically, then any specific application can realize its most frequent functions at the hard-wired level.

4. DECENTRALIZED, CONCURRENT CONTROL

Generalized sequential machines may perform not only primitive operations but also control other Gsm's. There are six basic ways in which one Gsm can control others:

hierarchical definition	the operations of one Gsm are given in terms of: the operations of other Gsm's.
logical dependence	...N Gsm's in a pipeline, each performing one of N sequential suboperations.
logical independence	...N Gsm's in a vector, each performing one of N parallel subprocesses.
logical separation	...N Gsm's in a vector, each performing one of N parallel subprocesses; however, the subprocesses are not totally independent.
repetition in control	...N Gsm's in a vector, each performing one of N parallel subprocesses; however, the desired result is not a combination of results from every subprocess but only from one (as in conditional or iterative statements).
repetition in data	...N Gsm's in a vector, each using one of N subitems as input to a given process.

Each of these methods of controlling other Gsm's is representable as a single item. In other words, these kinds of parallel processes can be written in a linear notation which contains the definition and state of each process. Let a Gsm be abstractly written as a pair of parentheses, representing its encoding, and a marker, ©, representing the current state of the Gsm.

(... © ...)

A hierarchical relationship between two Gsm's is written by including the inferior in the superior.

$$(\dots \odot \dots (\dots \odot \dots) \dots)$$

Control over other Gsm's is written by including them all as inferiors.

$$(\dots \odot \dots (\dots \odot \dots) \quad (\dots \odot \dots) \quad (\dots \odot \dots) \dots)$$

An expression in this notation can progress from one configuration to the next due to the actions of one Recursive Machine element or many. This means that the number of processors which actually execute a collection of parallel tasks needn't be decided beforehand. In fact, it can vary from microsecond to microsecond.

This approach to parallel processing has the advantages that:

- many small processing elements or a few middling elements or one large element are all controlled by exactly the same mechanism, which accommodates increasing miniaturization;
- movement of data is largely between bottom-level Gsm encodings; since these are contiguous, off-chip communication is minimized, which accommodates low-bandwidth pins;
- off-chip communication largely goes to adjacent chips, reducing system-wide communication;
- there is essentially no centralized control; at worst, a top-level Gsm may coordinate many large inferiors, and because of the items' great size, messages would pass across the entire system, but the higher-level busses in a RM are there specifically for low-frequency, top-level coordination.

4.1 Communication

Any function which is described in terms of more than one Gsm needs to tell not only how the individual Gsm's operate but also how they cooperate. Inputs and outputs of one are identified with inputs and outputs of others.

4.1.1 Hierarchical definition

Subroutines are functions which appear in the description of others, and which, consequently, send their outputs to whichever item sent them inputs. The addresses of these items are generally known in advance, but needn't be. Subroutining can be accomplished by augmenting the set of inputs with the address of the sender, and then sending the outputs to subitems at that address. Subroutines which appear in the description of only one other function can be physically located within the other's description.

$$(\dots \odot \dots (\dots \odot \dots) \dots)$$

4.1.2 Logical dependence

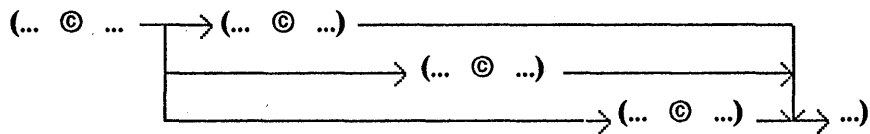
Sequential algorithms are composed of logically dependent functions. In terms of the abstract notation, logical dependence means that all outputs flow from left to right and that some outputs flow between inferiors.

$$(\dots \odot \dots \longrightarrow (\dots \odot \dots) \longrightarrow (\dots \odot \dots) \longrightarrow (\dots \odot \dots) \longrightarrow \dots)$$

4.1.3 Logical independence

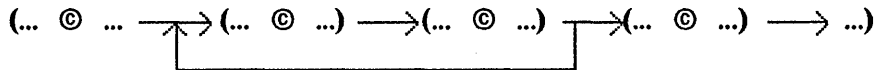
A restricted class of parallel algorithms are composed of logically independent functions. In terms of the abstract notation, logical independence means that all outputs flow from left to right and that

no outputs flow between inferiors.



4.1.4 Logical separation

A wider class of parallel algorithms are composed of logically dependent functions. In terms of the abstract notation, this means that most outputs flow from left to right but some do not.

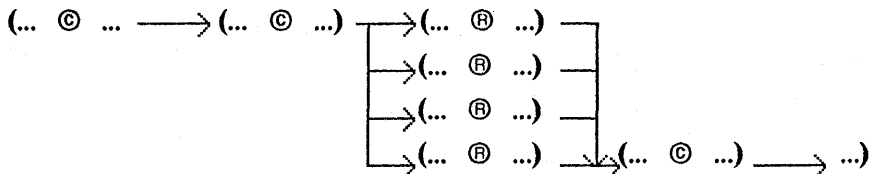


4.1.5 Repetition in control

Algorithms which derive their parallelism from their own structure are composed of functions which are selectively executed. In terms of the abstract notation, there is no characterizing pattern of cooperation; all possible flows may occur.

4.1.6 Repetition in data

Algorithms which derive their parallelism from the data on which they operate are composed of multiple copies of the same function, along with two bracketing functions which disperse and collect the data, respectively. In terms of the abstract notation, repetition in data means that the middle N functions are identical. An important point is that the number of replications, N, may be governed either by the structure of the data or by the structure of the activity being modelled or by any other extrinsic consideration.



4.2 Data-Driven Versus Control-Driven

Data-driven and control-driven models of computation are not mutually exclusive. Recursive Machine architecture supports both, allowing the choice between them to depend on the application.

Some functions are naturally driven by the availability of input data. In their descriptions, inputs are encoded as empty items. Functions which supply these values have their outputs encoded as "write" messages addressed to the corresponding inputs.

Other functions are naturally driven by the need for input data. In their descriptions, inputs are encoded as "read" messages addressed to the corresponding output.

5. RECURSIVE SYSTEM ORGANIZATION

5.1 System Organization

In order to survive several orders of magnitude more miniaturization, to allow systems to be configured out of a variable number of elements, and to allow configurations to tolerate failed elements, a system of elements must be isomorphic to a single element. That is, the method of interconnecting elements and the method of communicating between elements must be isomorphic at all levels of system construction.

A single element is partly a shift register onto which a logical data organization is mapped. Therefore, a system of elements must form a shift register. Each element, and recursively, each system of elements, must have two nearest-neighbor connections, over which quats are moved when operations force them out of any given element.

A single element has several external connections, over which information is exchanged with other systems. Therefore, a system of elements must have several external connections. Recall (from §2.4.3 Logical-to-physical mapping) that each element keeps track of the absolute addresses of the first and last quats in its portion of the overall shift register. Likewise, a system of elements has identical addressing logic which keeps track of the absolute addresses of the first and last quats in its portion.

A *Recursive Machine system*, then, consists of a number of RM elements or, recursively, a number of RM systems, which are interconnected in a chain and which have their external connections bussed together. Quats which pass in and out of both ends are monitored and the logical addresses of the first and last quats in the system are kept up to date. Any message which appears on the system's bus and addresses a nonlocal item is routed to the next higher bus. Conversely, any message which appears on the next higher bus and addresses a local item is routed to the system's bus.

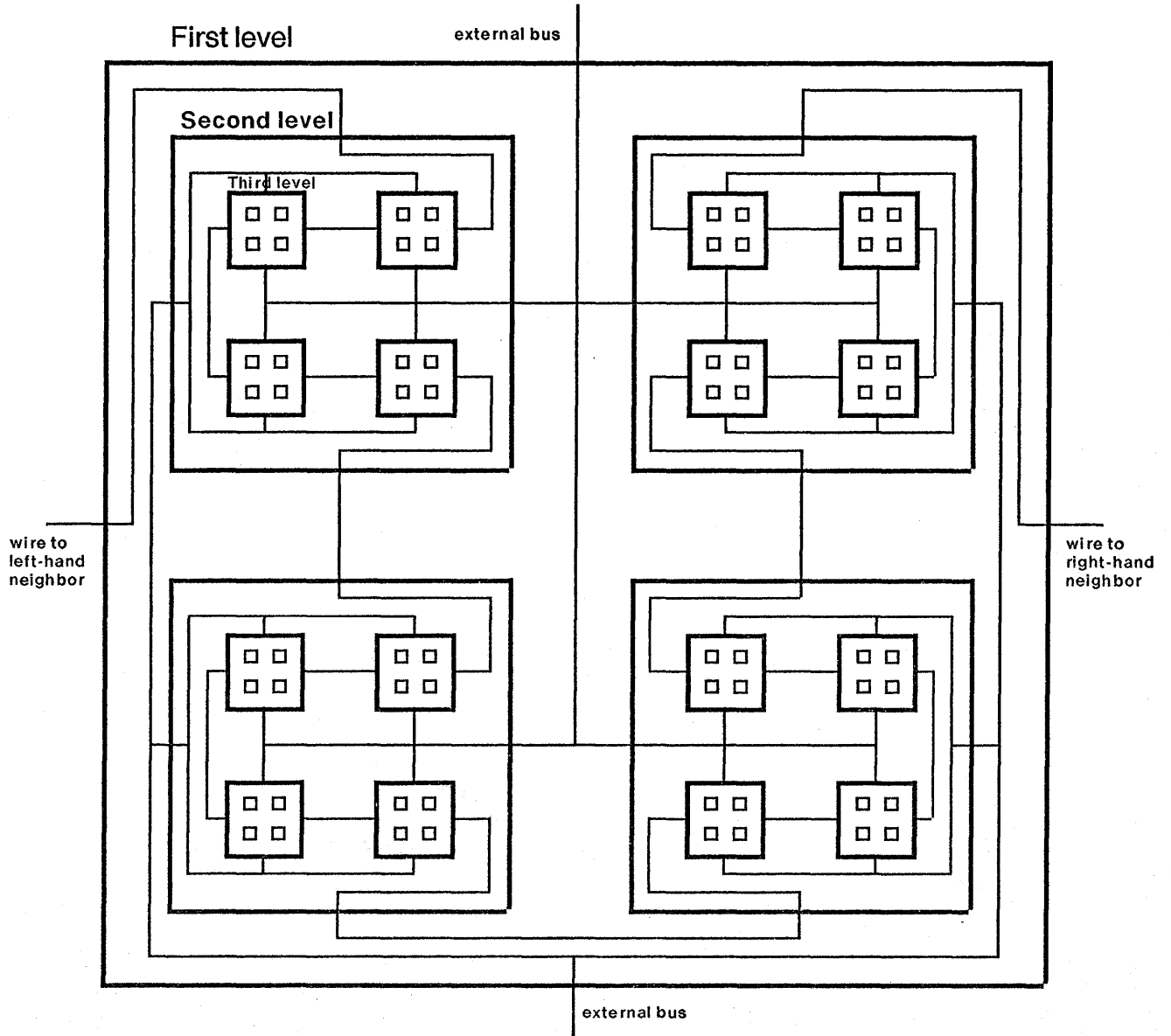
In the diagram on the next page, the outermost border surrounds a single Recursive Machine element, which is composed, recursively, of four RM's. Each of these is, in turn, composed of four RM's, and each of these, too, is composed of four RM's. There are $4^3 = 64$ elements at the lowest level shown. The interconnecting busses at this fourth level of system structure are not shown, nor are the bus coupling elements. Also, the physical diagram shows two busses, while the logical diagram shows only one.

[Implementation note: notice that any of the levels could correspond to chip packages. Recursive Machinery can be implemented with one element per chip in 1978, four elements per chip in 1980, sixteen in '82, 64 in '84, 1024 in '88. In other words, the system in the diagram can be implemented on one 100-package board today, and on $1/16^{th}$ of a chip ten years from now.]

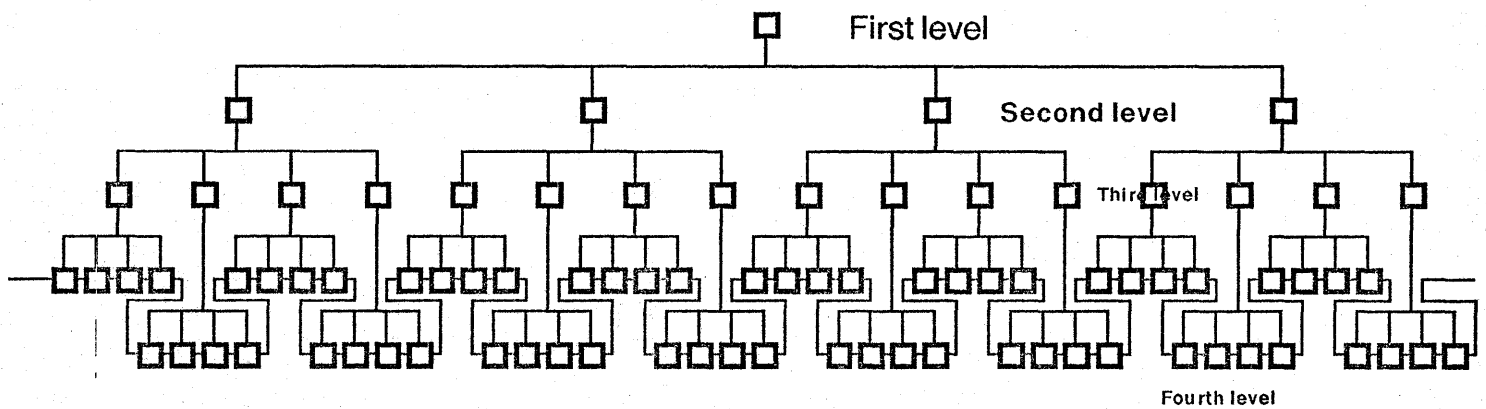
[Implementation note: advantage can be taken of the two-dimensional arrangement of RM elements to avoid having to store matrices in either row or column order. Instead, a matrix can be stored as a vector of quadrants, each of which may fall into a rectangular set of elements. Then, operations on submatrices can proceed in parallel.]

5.1.1 System intercommunication

We have already described one form of communication (in §2.4 Addressing) as a sequence of items, alternating between selectors and codes which determine the type of the following selector. All Recursive Machine communications are a simple generalization: the "type-selector" pair of an address is a special case; the more general case allows n-ary sequences of items in which the first item determines how to interpret the remainder. For example, the first item could be an addition operator, indicating that the next item is an addend, to be arithmetically combined with the recipient's value. Each element must listen to all messages and interpret all addressing operators to determine whether or not the remainder of the message must be routed to an item within its store.



Logical system organization:



5.1.2 Nearest-neighbor interconnection [Implementation note]

After considering the ultimate uses of information processing systems, the next most important consideration is the number of pins needed to interconnect system elements. What is the minimum needed to interconnect nearest-neighbors? Quats can be transferred one bit at a time. As to control lines, two cases may be considered. In general, elements may not be implemented identically; on the other hand, neighboring elements may be adjacent on a single chip. In either case, notice that unused space buffer other operations from the transfer of quats to a neighbor. Hence, we can implement the function with self-timed logic, using two control lines for "request to send" and "ok to send".

Elements which have a neighbor in an adjacent system do not require more pins for the system's addressing logic. Their chain connection simply fans out to it.

Since a single element can be an entire Recursive Machine, the nearest-neighbor connection makes the shift register circular. This is extremely desirable in multi-element systems, too, as a protection against failure of the first element. Reliability is increased if all elements are able to hold the root.

5.1.3 System interconnection [Implementation note]

What is the minimum number of pins needed to make external connections? One line is needed for data. The rest depends on the choice of bus protocol. Let us assume that synchronous operation is possible. Nearly all RM busses are local to a small area on a single chip. Higher-level busses interconnect similarly packaged systems. Therefore, a centrally arbitrated bus is admissible, using three control lines for "bus request", "bus grant", and "data present".

5.1.4 Total interconnection [Implementation note]

Using the figures of the two preceding sections, non-IO elements can be constructed with only $2 \times 3 + 2 \times 4 = 14$ pins, plus overhead.

Since this number is surprisingly small, implementors may consider wider connections. Data could be transferred in quats rather than bits, using an extra data lines per connection; this raises the count by four, to 18+. This is still small, so there could be more than two busses: four busses require 28+ pins; five, 33+; six, 38+. Alternatively, there could be larger data characters than quats: nine bits per character and two busses require 46+ pins.

5.2 Input/Output Organization

External devices are assigned to items with specific names. Sending "read" messages to the items assigned to input devices and "write" messages to the items assigned to output devices distinguishes information which is to be exchanged with the outside world.

Devices are physically connected to controllers which, on the one side, follow the device interface, and on the other side, follow the Recursive Machine element interface. That is, controllers appear to other elements to contain items, grab messages addressed to their items, and allow quats to flow in and out of their sides.

A major difficulty involved in doing IO is that, while items are normally free to migrate completely around the system-wide shift register, an IO item may not migrate out of the controller which is physically connected to its associated device. A major relief for this difficulty is to implement Recursive Machine operations so that there is *no net migration*; the time-average physical position of any item approaches a constant. This is not a problem; the obvious implementations have this property.

It can happen that a device transmits information which is many times larger than the capacity of the element to which it is connected...most of the information will have to emigrate to neighboring elements. This presents no problem for the given device, but if a controller has more than one device attached to it, the others' associated IO items may be forced out. A simple approach is to

attach each device to a number of adjacent elements, so that the associated item can migrate between them. Another relief is to bracket IO controllers with elements that have large stores and can buffer large transmissions. This reduces displacements into the rest of the system.

[Implementation note: it is undesirable to keep IO items at one end of the overall shift register. In general, the shift register is circular, and the ends may migrate. In 1979, when Recursive Machines have few elements, the end of the shift register is likely to remain in exactly one element; its failure would be catastrophic. In 1989, when Recursive Machines have tens of thousands of elements, external connections will necessarily run to elements which are separated by hundreds of others, in effect, distributing IO items throughout the system.]

5.3 System Initialization

Systems which consist entirely of volatile storage contain only unused space when they are turned on. The very minimum action to be taken is to create a root item in exactly one of the elements, since all RM operations are performed by some item.

Systems which contain nonvolatile storage can include a representation of a Gsm which accepts a power-on input and boots the system.

5.4 Error Recovery

Recursive Machines are very sensitive to failure in three places: (1) the continuity of the system-wide shift register; (2) delimiter balance, both in storage and during transmission; and (3) bus couplers, with increasing sensitivity at increasing levels.

The complete failure of an element destroys an arbitrary sequence of quats, since element boundaries do not necessarily correspond to item boundaries. However, the address information in elements on both sides of the failure allow the restoration of delimiter balance. Because groups of elements exchange information on busses, an element which drops off the busses does not affect the others. The shift register is more sensitive. It is necessary to provide bypasses around each element, which are activated when an element fails.

Considering delimiter balance during transmission, single bit errors result in syntactic nonsense:

original	erroneous	consequence
(0/1	extra) or (embedded in data
)	0/1	extra (or) embedded in data
()	(embedded in data or failure to terminate
)	() embedded in data or failure to terminate
0/1	((embedded in data or failure to terminate
0/1)) embedded in data or extra quats

Except for the termination failures, these syntactic errors can stimulate retransmission. [Implementation note: termination failures can be reduced by suffixing one or more go-quats to each message. If go-quats are lost in transmission, the superfluous ones make up the difference, allowing the receiver to detect an end to the message. The receiver can then count the received suffix to determine whether or not go-quats were lost. The advantage of suffixed go-quats is that a common kind of transient bus failure can be detected quickly with low overhead. Their disadvantage is that they do not make the probability of termination failure zero, so time-out is still needed.]

The complete failure of a bus coupler cuts off external access to a portion of the tree. Since the system-wide shift register does not go through bus couplers, information in the afflicted subtree can be recovered by forcing it to migrate into elements which can be accessed externally. [Implementation note: it may be tempting to attach devices to the highest bus coupler, which can access the whole tree, but the failure of that coupler would be catastrophic.]

5.4.1 Gsm errors

Since there will be logical errors made when constructing Gsm's, and since arbitrary Gsm's can be executed at the hardware level, the basic machinery aborts Gsm's which receive unanticipated inputs, which cycle without end, or which reach a terminal state. A Gsm which is aborted leaves a representation of its complete state at the time of the error, facilitating diagnosis. It is in principle easy to alter the representation of an aborted Gsm, either to modify its definition or to modify its state, to obtain a new encoding which will not fail as the old one did.

5.4.2 Robustness

Recursive Machines are insensitive to certain failures. All elements are anonymous and there is no way to tell in which element any particular operation is performed, so the failure of one element does not hamper the execution of programs by the remaining elements.

Transient failures often precede hard failures. Elements which detect transients can flush their stores, leaving only unused space, and activate the chain bypass, effectively taking themselves out of the system before any information is lost. Such graceful failure will be vital when there are over 1000 elements per chip, since it may not be practical to replace chips while 900 or 800 elements are still functioning.

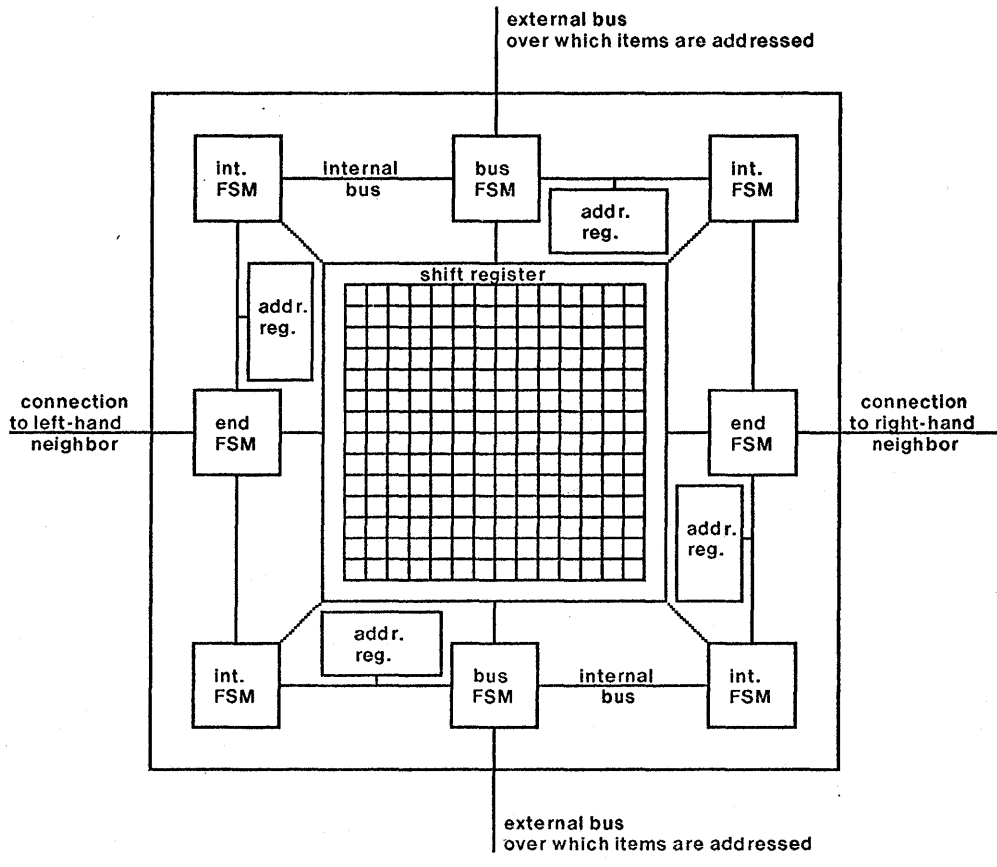
Systems with many elements will not have all of them executing instructions at every instant. When idle, RM elements can verify that the delimiter balance within their own stores agrees with their addressing information, adding a degree of reliability.

6. RECURSIVE MACHINE ORGANIZATION

In §5 we saw that the principle of Recursive System Organization requires a single RM element to have an interface which is isomorphic to that of a system of RM elements. One of the harmonies between the RM principles is that this interface is all that is needed to fulfill the other principles. The vital parts of a Recursive Machine element are derivable mainly from this interface, namely:

- a shift register, over which items may be distributed in space, monitored at each end and at several interior points by finite state machines, or Fsm's;
- external transmission media, over which items may be distributed in time, monitored by a Fsm; one medium is not sufficiently reliable to be adequate in Recursive Machines;
- address registers, which track the absolute addresses of:
 - the first and last quat in the shift register, and
 - one or more interior quats which are to be manipulated [although it may be possible to have just one address register];
- Fsm's to evaluate a certain encoding of Gsm's, having:
 - logic to evaluate all possible encodings (a universal Gsm interpreter);
 - logic to evaluate certain frequently needed Gsm's (primitive machine operations);
 - programmable logic, to allow arbitrary encodings to become part of the hardware dynamically;
 - logic to convert encodings into programmable logic form (these latter two are required by the principle of Recursive Machine Language);
- one or more internal transmission media, connecting the various parts.

The interior Fsm's interpret instructions; the end Fsm's pass quats between adjacent elements; bus Fsm's select messages intended for items within the element. Although eight Fsm's are shown in the adjacent diagram, a Recursive Machine element generally has at least one Fsm per port and one or more interior Fsm's.



All Fsm's may operate at once. The bus Fsm's can be performing address comparisons while both neighbors are passing quats to the end Fsm's, and while each interior Fsm is interpreting a Gsm encoding. [Implementation note: this argues for dividing the shift register into several sections so that several Fsm's can be accessing different parts of it.]

[Implementation note: in order for a single element to be truly all-purpose, it must support a standard interface as well, such as RS-232C.]

7. PROGRAM EXAMPLE

Dijkstra's Dutch National Flag problem is a simple function which involves sorting, naming, selection, iteration and arrays. Given an array of colored stripes, $C [1:N]$, arrange the stripes so that all the red ones occupy the lowest positions in the array, the white ones the middle positions, and the blue ones the highest positions. Each stripe may be examined, only once, by using a function *color* (i) which returns the color of the stripe in position i . In the original problem, the only way to manipulate the array was by means of a function *swap* (i,j) which interchanged the stripes in positions i and j .

Given that Recursive Machinery provides an arbitrary number of generalized sequential machines for the implementation of algorithms, we can consider several mental models of the computation which are automatically rejected by von Neumann architecture. One image is that of having nearest-neighbors exchange stripes. For example, any item holding a red stripe which was to the right of an item not holding a red stripe would trade with its neighbor. If all items were to perform this operation, all red stripes would be traded until they occupied the left-most items. We, too, have to reject this model because it violates the restriction that each stripe be examined for color only once.

Another unusual model can be considered. If it happens that colors can be ordered, and that red < white < blue, then the array can be sorted by copying it into an item whose subitems are to be *ordered*. As each stripe is put into the new one, its position will be chosen by the hardware so that ordering is preserved. All red stripes will be put at the left end, blue at the right end, and white in between. This very simple way to sort is an inherent advantage of Recursive Machinery. Notice that it does not require additional storage since each item is not copied but literally moved. For the Dutch National Flag problem, though, it amounts to tricky coding. The technique does not work for arbitrary color arrangements, so we reject it, too.

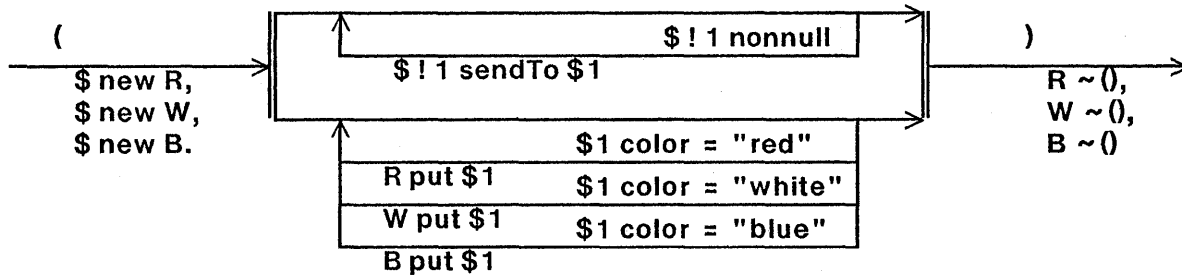
The natural way to sort on RM's causes us to think of the stripes as distributed in time, *i.e.*, as a stream, rather than in space as an array. Suppose we direct the array C , as a Gsm, to transfer each stripe on the basis of its color to one of three other Gsm's named Red, White and Blue. After all stripes have been transferred, the three Gsm's can concatenate their contents to form the Flag.

One way of implementing this model is to have the array C add three empty items to itself. It then sends its original contents to itself, distributing the stripes to the three new items according to color. At the end, C contains only the three items. The final operation is to remove one pair of parentheses from around these items, leaving the sorted stripes.

To write this algorithm down, let us indicate concurrent, subordinate Gsm's in railroad notation by T-intersections with parallel lines, directing the main-line toward the initial states of the subordinates and from the final states of the subordinates to the continuation of the main-line. This notation has the same semantics in any of the following three possible situations: (1) C is wholly contained in one RM element which has one Fsm available to control the main-line Gsm and each subordinate Gsm; (2) C is wholly contained in one RM element which has enough Fsm's available to control each Gsm concurrently; and (3) C is split across several RM elements and each uses a Fsm of its own to control one of the Gsm's.

In the diagram below, the color function is written in message form: Addressee "color" Parameters. We use the notation:

"\$"	to signify an item itself;
"\$n"	to signify the nth input of the item;
"\$! n"	to signify the nth subitem;
"new X"	to signify the creation of a subitem named X;
"= X"	to signify testing an item for equality with X;
"sendTo X"	to signify removing an item to X;
"~()"	to signify exploding an item into its subitems (for example, "((A)(B)(C)) ~()" yields "(A) (B) (C)", distributed in either space or time).



Notice:

- delimiters make the array bound N superfluous;
- if there are no stripes of a given color, the corresponding collector receives and sends no items;
- if there are stripes of a wrong color, the system enters a default error state; another response could be attained by explicitly acknowledging wrong colors;
- if transmission of the inputs is prematurely terminated, the machinery will be hung on input...it can be cleared by supplying a closing delimiter, which will reveal what inputs were received;
- if transmission of the outputs is prematurely terminated, the machinery will be hung on output...it can be cleared by acknowledging its remaining outputs.

When this function is programmed for traditional machinery, one must carefully manipulate several pointers into the array. It is all too easy to make logical errors. Recursive Machinery provides alternate models of information and processing which are rich enough to eliminate programming intricacies at this level.

8. PERSPECTIVE

8.1 Experience

Recursive Machines have benefitted from these lessons of traditional architecture:

- ◇ The expressiveness of our programming languages, and hence our ability to map problems into algorithms, is limited by the basic capabilities of our hardware. In order to serve human organizations which are highly concurrent, dynamically restructuring themselves, and dynamically redefining their activities, hardware must possess these same characteristics.
- ◇ Convoluting machines engender convoluted programs.
- ◇ Most problems have to be overspecified in order to become programmable, most often having excessive sequentiality or excessive data ordering.
- ◇ There must be a deep harmony between data structure and program structure. Object-oriented systems exhibit an alternate harmony to that of von Neumann computers.
- ◇ Managing storage can dominate machine activity if not done right. It has been hard to overlap allocation and deallocation with normal access. It has been hard to map names which are convenient at the software level into names which are convenient at the hardware level.
- ◇ Applications always run out of address space.
- ◇ Given a hardware representation of numbers which is precise to n decimal places, applications eventually need $n+1$.
- ◇ Microprograms always run out of control store.

- ◇ Tagged architectures always run out of tag space.
- ◇ Descriptor-based machines can avoid 70% of the data movement which takes place in machines which lack descriptors.
- ◇ Byte-oriented computers require 100% more storage, on average, than bit-oriented computers.
- ◇ Important flexibility is gained by moving information about the structure of data out of programs and into data structures.
- ◇ Important simplifications in programs have resulted from bringing input/output into the main address space, from bringing secondary storage into the main address space, from bringing process state into the main address space.
- ◇ There is always one instruction which a machine's users would retard in order to speed up their most frequent subroutine.
- ◇ Contemporary bandwidth requirements are too high for system-wide busses.
- ◇ Protection mechanisms must be allowed for in the hardware primitives.

Note that the principles of von Neumann architecture cause of many of these problems.

8.2 Anticipation

In addition, Recursive Machines accommodate the changes which VLSI brings.

- ◇ Basic machine elements will have limited access to the rest of the system, prohibiting centralized resources and centralized control.
- ◇ Machine elements must survive three orders of magnitude more miniaturization, necessitating extreme regularity.
- ◇ Logic-poor systems will no longer have a general advantage over logic-rich systems.

Note that von Neumann architecture makes none of these accommodations.

8.3 Objective

The ultimate claim is that an architecture which fulfills these observations is well suited to office information-processing.

8.3.1 Office information

Recursive Machines model office information well by explicitly representing structure. For example, filing is modelled directly in terms of a top-level item which represents a room of filing cabinets, whose subitems represent cabinets, whose subitems represent drawers, whose subitems represent folders, and so on through sets of documents, documents, pages, lines, words, and characters. Any operation which the hardware can perform on character strings (such as move, copy, and delete), it can also perform on any higher-level structure. Furthermore, if each level is distinguished from others by a descriptive tag, rather than by fixing the interpretation of depth, then it is possible to create and destroy new levels in the hierarchy without having to rewrite existing information.

Xerox's IISS model of information as a value with 15 attributes is easily represented by an item consisting of 16 pairs, where the first of a pair is an encoding of the kind of attribute (i.e., value, name, description, length, format, responsibility, standardization level, security, status, alias, synonym, algorithms, characteristic indicators, coding/edit rules, coding list, standard header) and the second is the value of the attribute. Since all the attributes are part of a single superior item, they can be copied and moved as a unit. Access to any particular attribute is done by addressing by content to find the first of a "kind, value" pair and then by addressing self-relatively (namely, "next") to obtain the value of the attribute.

8.3.2 Office organization

Recursive Machines model office organizations well, by having a physical hierarchical structure and by directly supporting logical hierarchical structures, with no predefined limits on the depth of hierarchy nor on the branching at any node, as well as allowing hierarchies to change dynamically. An ordered set of items corresponds to an assembly line. An unordered set of identical items corresponds to a pool. An unordered set of dissimilar items corresponds to a bull-pen.

8.3.3 Office communication

Recursive Machines model office communication well, by having a total order among all communicators, by allowing any sort of message to be initiated by any communicator, and by having hierarchical communication lines. In any company with thousands of employees, hundreds of small groups can talk simultaneously in offices and hallways, while more distant couples are talking over the phone, while individuals are sending out memos to hundreds of others or addressing a roomful of colleagues. There is a distinction to be recognized between media (*e.g.*, rooms, hallways, telephones) and agents (employees). In a Recursive Machine, all these modes of communication can take place, and simultaneously. Elements and their interconnections are media; items are agents.

8.3.4 Office procedures

Recursive Machines model office procedures well. The RM element functions as someone in an office who is available to perform whatever procedures are naturally located at that office. Arriving inputs are like phone calls, letters, and ordinary conversation, providing new information for which various procedures may have been waiting. They interrupt the element and either cause it to resume working on a particular task or are queued until the task to which they contribute is resumed for other reasons.

9. REFERENCES

- [Davis] Davis, A. L., "The Architecture and System Method of DDM1: a Recursively Structured Data Driven Machine", *Proc. Fifth Annual Workshop on Computer Architecture*, 1978, pp. 210-215.
- [Kain] Kain, R. Y., *Automata Theory: Machines and Languages*, McGraw-Hill, 1972.
- [Kay] Kay, A., "Microelectronics and the Personal Computer", *Scientific American*, September 1977, pp. 230-244.

For Xerox internal use only

Whole ALTO World Newsletter

Technology and Tools

XEROX

June 6, 1979

SPECIAL ANNOUNCEMENTS

WHOLE ALTO WORLD MEETING

The next Whole Alto World meeting is coming up soon. It will be held in the El Segundo area and hosted by Ted Davis and PD. Here are the details:

Tuesday, June 19, 1979 - 8:30 AM to 4:00 PM

Airport Hyatt House, The *Mikado Room*, Sepulveda at Century Blvd. We will have coffee and pastries in the morning; luncheon arrangements have been made.

Attached to the Newsletter is a map giving further details about the location of the Airport Hyatt House.

GENERAL NOTES

A CHANGE AT THE HELM

As most of you already know, Frank Ludolph has left WAW for new duties in SDD. Those of us in WAW would like to thank Frank for the outstanding contributions he made as Coordinator, particularly during our formative period.

As of June 1, a new WAW Coordinator is on board. He is Ron Cude, now a member of the *ASD Field Support Group*. Ron comes to *ASD* and WAW from the *Special Programs Manufacturing Group* (Alto purveyors to the WAW) where he was Test Manager. Ron will be serving as a information contact for all Alto users so be sure to contact him when you have Alto-related questions. Here's where:

Xerox Corporation
701 S. Aviation Blvd.
Attn: Ron Cude, A3-83
El Segundo, Ca. 90245
Intelnet 8*823-2465
Message: <CUDE>

The function of the Whole Alto World Coordinator is to provide Alto users and maintainers with information, a focal point for communications, and to facilitate the transfer of technology within Xerox and it's Alto community.

THE NEWSLETTER

The Newsletter Editor would like to invite all Alto users to utilize this space for any item you think may be of interest to others in the Alto World. Keep others informed of your activities, your thoughts regarding anything Alto-related, new software, new hardware, etc. All of these and more are of interest to us all in the rapidly changing environment we share. One area that can be of particular usefulness is the *MARKET PLACE* section. Let the Newsletter help you get in touch

Whole ALTO World Newsletter

with others sharing a common interest be it in hardware, software, graphics work, administration, games or ????. Be sure to have anyone who would like to receive the Newsletter, but currently is not, message or call Ron to get on the mailing list. If we get the approval of the majority of the people at the next WAW meeting, we will be mailing the Newsletter only to those of you who do not have message accounts. People with message accounts will be sent a message reporting the publication of a new Newsletter.

MARKET PLACE

Market Place provides a forum for Alto users to make offerings and requests for Alto-related hardware and software. To place an "ad" or offering, send the text to the Coordinator, Ron Cude, at El Segundo. Message <CUDE>, or phone Intelnet 8*823-2465.

TOOLS

HARDWARE

SPG is still cranking Alto's out at a rate of eleven per week. In progress now is the tenth build. Who ever thought it would go this far?

Have you built a special purpose interface for your Alto? Why not let WAW know about it. There's no use having others "reinvent the wheel" when it isn't necessary. You may be surprised what good stuff is already available, precluding your having to tie up resources and money on a duplicate (or almost so) project.

DOCUMENTATION

New Hardware Manual - The Alto Hardware Manual has been revised to describe the forthcoming 3KControl RAM option for Alto II's and to incorporate a few other minor changes. See [MAXC]<AltoDocs>AltoHardware.press.

Alto Sub-Systems Catalog - The catalog is in the process of being updated by Bruce Nelson of CSL and will be included in next month's Newsletter.

MAINTENANCE NOTES

Hey Maintainers! This section is meant primarily for you. As new Alto changes or additions happen, this section will tell you all you should know about them (I hope). If any of you come up with any special trouble-shooting techniques, unique problems or whatever you think might interest others like you, why not let WAW tell everyone else about them. Working on something for hours is no fun when a little sharing of inside tips might cut your down time to only minutes.

The following was provided by Dave Damouth at WRC:

"Recent open heart surgery on my mouse was a success. I had one of the "bad" ball mice, which did not roll smoothly, and felt like the ball had flat spots. I removed the mechanical assembly, and removed a general accumulation of dust and lint by wiping exposed surfaces with alcohol. Klaus Stange flushed the ball bearings with light oil, forcing it through them and removing the excess with air pressure. After reassembly, it worked very smoothly. This is a quick and easy procedure, providing that the delicate commutator wipers are treated with respect."

Whole ALTO World Newsletter

This is a good procedure as long as extreme care is taken while your Mouse's innards are exposed. Also, in oiling the bearings, be sure to use **ONLY** light gun oil or an equivalent. Anything else may render them virtually useless. As an added note, *SPG Manufacturing* is now gearing up to repair Mice. As with other field-returns, there is a maximum four week turn around on them. If you have a sluggish or inoperable Mouse, plug in a spare and try Dave's procedure or send the poor critter to SPG, attention: Tom Logan, M1-38, for an operation.

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local File Server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [ISIS].

ReReleases - Subsystems

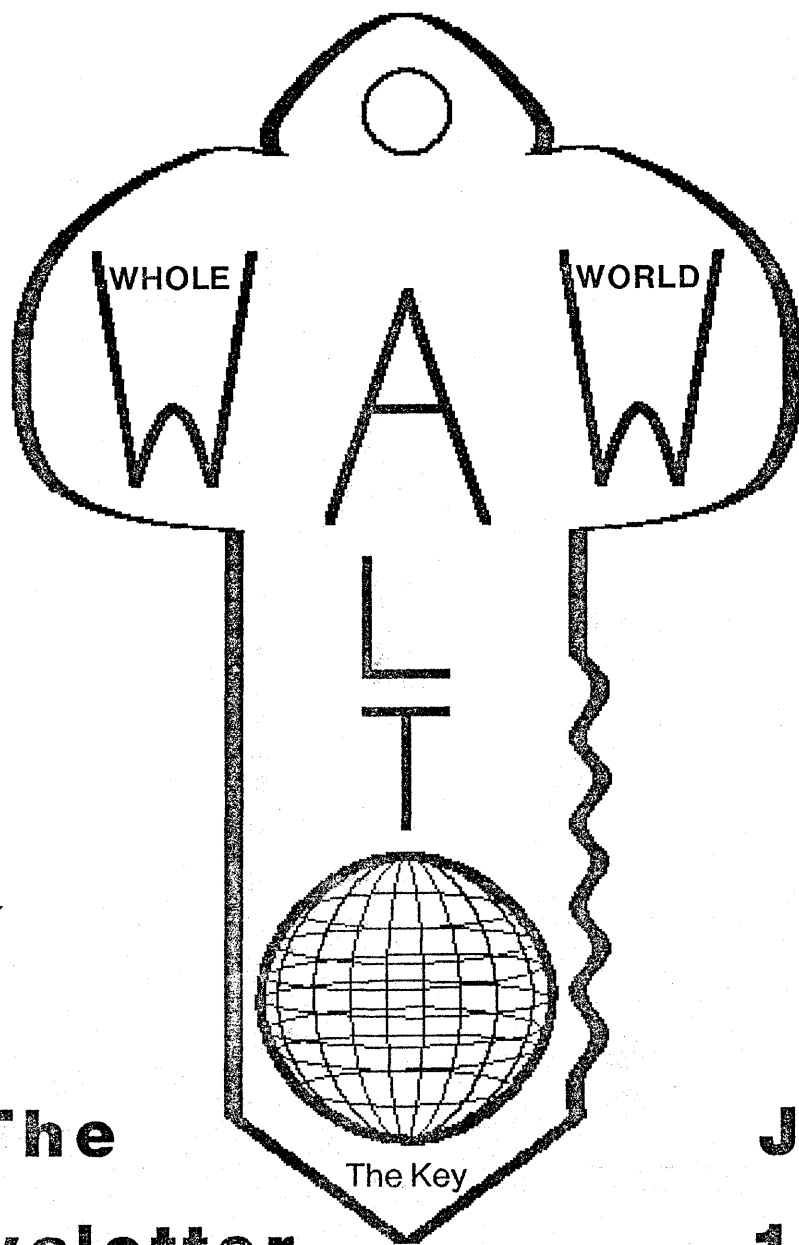
SIL - "Expand mode" now expands the font as well as lines. This could be especially helpful when using the Template font to see when Template characters and lines line up correctly. ↑E expands the rectangular area defined by the two preceding marks so that it fills the screen (as nearly as possible). The magnification factor (from 2 to 9) is shown in the status line. A second ↑E exits the mode. The sense of ↑X and ↑shX are reversed when in magnified mode. Retrieve <ALTO>Sil.Run.

ANALYZE - A couple of seldom exercised routines of Analyze have been found to be at fault. If you have had problems recognizing dictionary names from User.Cm or use font 1 italics INSIDE your library definitions, then you should get the latest Analyze. Retrieve <ALTO>Analyze.Run.

TECHNOLOGY

Featured each month will be articles and papers on technologies affecting or affected by Altos, new technologies and directions developing within Xerox, and discussions of the work-in-progress within specific organizations. Please send anything along these lines to Ron Cude for inclusion within the Newsletter.

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC] and [OLY]<AltoDocs>WawNewsM-YY.press or may be obtained from the editor, Ron Cude, ASD, by messaging <CUDE> or calling Intelnet 8*823-2465.



**The
Newsletter**

**July
1979**

XEROX

Whole ALTO World Newsletter

For Xerox internal use only

CONTENTS

SPECIAL ANNOUNCEMENTS	1
GENERAL NOTES.....	1
Alto National Network Topology	51
Alto Network Map	52
MARKET PLACE.....	3
Complete Alto Drawings List.....	53
TOOLS.....	3
Hardware	3
Maintenance Notes.....	3
Software.....	4
Re-Releases - SubSystems	4
How To Use IFS.....	26
Re-Releases - Packages.....	6
TECHNOLOGY.....	6
CeBIT '79: A Report On The Office Equipment Show At The 1979 Hanover Fair.....	7
Report On The 1979 International Summer Consumer Electronics Show	13
A Report On The IWPA National Syntopian VII - Chicago.....	18
CATALOGS.....	6
Alto SubSystems Catalog.....	37

XEROX

Whole ALTO World Newsletter

For Xerox internal use only
Published and edited by Ron Cude
Message <CUDE> or call Internet 8,823-2465

TECHNOLOGY AND TOOLS FOR THE FUTURE

July 25, 1979

SPECIAL ANNOUNCEMENTS

WHOLE ALTO WORLD MEETING

Our next meeting will be held Tuesday, October 2, 1979 in Lexington, Massachusetts. Darwin Newton of Ginn & Co. will be hosting. Make plans now to attend. Next month's Newsletter will have further details. Please message or call Ron Cude if you think you will be attending. We will explore reserving a block of rooms at the Sheraton Lexington.

ALTO HARDWARE CATALOG

The Alto Hardware Catalog is being updated. I would like to take this opportunity to solicit inputs from anyone having an item qualifying for inclusion. Devices included in the Catalog must have at least one prototype considered to work properly and to be reproducible from existing documentation.

GENERAL NOTES

CHANGE IN NEWSLETTER

Attentive readers may have noticed some changes in the appearance of the Newsletter. Besides the cosmetics, it is now being created using BravoX. Any comments on the appearance or content of the Newsletter are always appreciated.

LAUREL FOR NON-LAUREL USERS

On May 21 the GSD Mail Center in El Segundo initiated an experimental Laurel Hardcopy service for non-Laurel Users.

This enables Alto users located virtually anywhere to send unclassified messages via Laurel to any Xerox employee in El Segundo by addressing the message to *Employee Name Mail Stop*<ESMail>. When using the "cc:" line, copy recipients in El Segundo may be identified by typing *Employee Name Mail Stop*. Please note, <ESMail> should not be shown here, otherwise only <ESMail> will appear on the hardcopy instead of the employee's name and mail stop.

Whole ALTO World Newsletter - July 24, 1979

For example:

To: Gene Diamand M4-04<ESMail>
cc: Rube Brady A1-50, John Lund A3-44, Wegbreit, ABC†

Each day, El Segundo Mail personnel periodically query Laurel for new messages, invoke the requisite "Hardcopy" command, and deliver the hardcopy messages. Names and locations on distribution lists containing El Segundo addresses should be visible.

If a message is undeliverable, the Mail Center will notify the sender specified in the "From:" line, via Laurel, with an explanation.

Due to the initial success of ESMail, by September the Mail Center plans to include the entire Los Angeles Basin. Messages sent via <ESMail> to other Xerox facilities in the greater Los Angeles area will be delivered through the Courier Network which services the majority of Xerox locations in the Basin. Delivery of messages will have a maximum overnight turnaround. A formal announcement will be forthcoming indicating the names and location codes of the facilities to be serviced.

Depending on user response and on work being done by ASD and CSL, plans for the future include:

- ▶ additional locations
- ▶ on request, confirmation of delivery
- ▶ automatic look up of the GSD INTELNET data base to immediately notify senders of undeliverable messages
- ▶ properly encoded classified messages

Please direct any comments to Jerry Torres at <ESMail>, El Segundo Mail Stop A1-50, or INTELNET 8*823-1169.

A CONVENIENT LAUREL HACK

The following was provided by Hugh Lauer:

I use the following hack as a convenient way to print copies of documents which are "distributed" via a message:

When an interesting document is announced, GET the file *Rem.Cm* into the Laurel message composition window. Add to the end of it the command: **HARDCOPY** <filename> <CR> (it is usually convenient to copy the filename, complete with server identification, directly from the message announcing the document.) Then **PUT** the contents of the message composition window back on *Rem.Cm*. Repeat this process as often as necessary while reading your mail with Laurel. At the end of the session, **QUIT** from Laurel and let the accumulated commands do the printing for you. To make this work, you need on your Laurel disk: *Empress.Run*, *HardCopy.Run*, and enough space to hold a reasonably sized document. One word of warning: Not all messages contain precisely the right file name e.g. typo's, misspellings, incorrect subdirectories, etc. A human rummaging around the appropriate directory can usually find the file he or she wants, this scheme cannot.

MY PRINTED DOCUMENT LOOKS FUNNY

If you have trouble printing Press files retrieved from [XEOS] (or elsewhere), it probably means you have been another victim of the great font kludge. Applying PressEdit to the file in question will add the time stamp and allow the file to print correctly. The only remaining question would be whether or not you can pleasingly print the "illuminated initials" some people use. *Contributed by Don Winter.*

Whole ALTO World Newsletter - July 24, 1979

THE EVER EXPANDING WORLD OF ETHER

The latest edition of the Alto Network and Network Map are included this month (see pages 51 & 52). Copies can also be found on [MAXC]<AltoDocs>AltoNetwork/NetworkMap.Press. Remember to message Art <Axelrod> to register your Alto, Gateway or Printer Server name.

MARKET PLACE

Market Place provides a forum for Alto users to make offerings of, and requests for, Alto related hardware and software. To place an "ad" or offering, send the text to the Editor, Ron Cude, at El Segundo, message <CUDE>, or phone Intelnet 8*823-2465.

ALTO HARDWARE DRAWING LIST

Appended to this month's issue (beginning on page 53) is a numerical listing, prepared by Carol Williams of SPG, of all Alto hardware drawings currently available and their latest revision levels. If you need a particular drawing, call Carol, Intelnet 8*823-1654 or message her via <CWILLIAMS>.

TOOLS

HARDWARE

★ ALTO II MAG TAPE PACKAGE ★

Alto II's can now run 1600 BPI 9-Track Phase Encoded (PE) Magnetic Tapes. An installation requires two ASD tape controller boards, cables, and a few backplane modifications. Both Kennedy (ANSI) and Mohawk style drives can be accommodated, with up to four drives daisy-chained by one controller set. The initial software has been released, which includes Tape (integrated with Trident) microcode, low level (buffer I/O) software, a stream package, a utility for copying unlabeled tapes of varying formats and a diagnostic. Interested parties should contact Alan Paeth, % Liz Bond, XEOS, MS: 359, Intelnet 8*844-2232.

MAINTENANCE NOTES

★ "G" REVISION CHANGE FOR ETHERNET MODULE ★

An Engineering Order has been officially released by SPG Engineering. It is applicable to Ethernet module's built before March 23, 1979 ("F" revision or earlier). The E. O. consists of changing four resistors to a new value and removing two others. Some Ethernet module's have exhibited marginal timing problems (particularly when talking to servers) and this change should cure them. A copy of the E. O. can be found on the last page of the Newsletter.

Whole ALTO World Newsletter - July 24, 1979**SOFTWARE**

In general, the subsystems, packages, and documentation indicated here will be available from your local File Server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [ISIS].

RE-RELEASES - SUBSYSTEMS**★ SMALLTALK CALENDAR ★**

A new version of Smalltalk Calendar is released with the boot files on [MAXC1]<Alto> Calendar.Boot and XMCalendar.Boot. This new release features the handling of multiple calendars with rapid switching between calendars via the integration of calendars with the Smalltalk project windows. This enables the user to point at a calendar and manipulate it without having to save/restore from Alto file representations of a calendar to switch from one to another as was previously required. Of course the linkage to saving and restoring to/from Alto files is retained for backup. The new release is also faster. This should be most noticeable to Alto I users. In particular, the menu comes up much faster thanks to a fix by Dan Ingalls. Also as a calendar gets cluttered with events which get finished, partially progressed, or cancelled from the past, it no longer slows down with the increased number of entries. However, these entries do still take up virtual memory space (hence DISK space); so, the user should prune them out periodically. The boot files have been reduced in size by 250 pages as some Smalltalk code which Smalltalk Calendar does not use has been trimmed out. A revised writeup is also available under <Altodocs> SmalltalkCalendar.Press

★ AIS VERSION 3.0 AND 3.1 ★

There is new AIS software available on [Maxc]<AIS> and [Erie]<AIS>. Following is a summary of the changes:

AIS.Run Version 3.0

1. Memory space was in short supply, so several features duplicated elsewhere were removed to make room for others. Those features removed were: Delete, Attributes, Annotate, Show, Halftone, Zoom. A copy of the code and sources of version 2.0 has been saved for posterity.
2. The 'Rotate' code was modified to work for both bit/pixel and byte/pixel images. The modifications were made by Brett Fleisch.
3. The "Press Printer Output to AIS" reformat software now handles Press.bits files that are in Orbit format. AisGetOrbitPage.bcpl was written by Bob Lyons.
4. The AIS printing software should now print from Tridents other than TP0.

AISUCA0.BR

1. An error in the calling arguments was fixed. The problem resulted in not being able to open files on Trident disks.
2. The minimum allowable stack size has been reduced to 500 words (if you dare!). The default size remains at 3000 words.

Whole ALTO World Newsletter - July 24, 1979*AIS.Run Version 3.1*

1. AisReformat - Can handle Press.Bits files wider than 3072 bits.
2. AisPrint - Sets the clock properly for printing grayscale pictures.

★ SIL ★

There has been one minor change to Sil. The "line pulling" feature of the Move command is now identical in expand and non-expand modes. There are also a couple of other hidden changes, and about 200 extra words of free space.

★ PRESSEEDIT 1.82 ★

Is now on [MAXC]<Alto>, and <Altosource>. It now understands rotated fonts. The EARS machinery has been removed. Complaints, etc. should be directed to Bob Lansford, XEOS, or message Bob in care of <LizBond>.

★ TFS & TFU ★

A new version of the BCPL Trident disk software is released. It includes some changes that have been made in preparation for the release of OS Version 17, notably extension of the DSK object to add generic operations for running the disk at the DoDiskCommand/GetCb level. This version of TFS will run under OS 16 also. However, the old TFS will NOT run under OS 17. Therefore, if you maintain software that used the TFS, you are advised to release a new version of that software loaded with the new TFS. OS 17 will probably be released in a month or so. (any source modules that get "Tfs.d" must be recompiled.) The revised documentation may be found as <AltoDocs>TFS.tty.

★ IFS 1.21 ★

Version 1.21 of the IFS software is released and is available on [Maxc1]<IFS>. The most important change is that a mail forwarding mechanism has been introduced. This will permit most non-Palo Alto mailboxes to be removed from Maxc and instead located at local IFSs. The necessary corresponding changes to Laurel have already been made and released in Laurel 5. The transfer of user mailboxes from Maxc to local IFSs will begin as soon as IFS 1.21 is in operation at all sites.

Other changes include direct transmission of Press files to printing servers; introduction of name, time, and boot server (probably not of interest to most IFS sites within Xerox); ownable user groups; and a number of minor changes.

The documentation in files [Maxc1]<IFS>HowToUse.press and Operation.press has been revised. It is suggested you retrieve and read these documents carefully before you install IFS 1.21 at your site. In particular, if you have been using the mail server feature in IFS 1.19, you should read carefully section 8 of "IFS Operation".

The new IFS has been running on Ivy, and though there have been several crashes, all have been accounted for; there are no known bugs. Some improvements have been made in memory utilization, and it is hoped that the "buffer deadlock" problem will be either eliminated or greatly reduced in frequency. (Another bug that would occasionally cause individual connections to hang has also been fixed.)

Those of you who modify the IFS software should know that the sources have been repackaged. A document now exists describing how the sources and command files are organized; see IFSSoftwareMaint.press.

Please let Ed Taft know if you encounter any problems. Thanks go to Dave Boggs and Steve Butterfield, who made substantial contributions to this release. A copy of How to Use IFS (Version 1.21) is included (page 26) within the Newsletter.

Whole ALTO World Newsletter - July 24, 1979**RE-RELEASES - PACKAGES****★ BCPL EFTP PACKAGE ★**

A new version of the Bcpl EFTP package is released. It has been divided into several pieces (Send, Receive, and Common) and repackaged, but the external interfaces are unchanged. One bug has been fixed. See the revised documentation in [MAXC] <AltoDocs> EFTPpackage.tty. Related to this, the extension to the EFTP protocol used to send files to printing servers is now documented. See [MAXC]<Pup>SpruceProtocols.Press. A few changes have been made in the interpretation of abort codes, so if you maintain any software that sends files to printers, you should read this document carefully.

★ BCPL PUP PACKAGE ★

A new version of the Bcpl Pup package is released. There have been a number of internal improvements, the most important of which is a modification to the routing table management algorithm to eliminate the upper limit on the number of networks (formerly 32). Since the number of networks in the Pup internet is approaching 32, maintainers of programs that use the Pup package are advised to release new versions that incorporate the new Pup package. The external interfaces to the new Pup package are upward-compatible from the old. Client programs that "get" any of the Pup declaration files should be recompiled. See the change summary at the end of <AltoDocs>PupPackage.tty for details.

TECHNOLOGY

Featured each month will be articles and papers on technologies affecting or affected by Altos, new technologies and directions developing within Xerox, and discussions of the work-in-progress within specific organizations.

IT'S BEEN SHOWTIME

This month the Newsletter brings you three reports dealing with recent shows. Two are co-authored by Dave Thornburg and Roy Lahr: "CEBIT '79: A Report on the Office Equipment Show at the 1979 Hanover Fair" (see page 7) and "Report on the 1979 International Summer Consumer Electronics Show" (see page 13). The third, by Ginger Engstrom, (see page 18) is titled "IWPA National Syntopian VII - Chicago".

CATALOGS**ALTO SUBSYSTEMS CATALOG**

The Alto SubSystems Catalog has been updated and revised. The Catalog contains an alphabetical listing of many commonly used Alto subsystems and a short narrative about each. If one of your favorite subsystems has been left out, let me hear about it and we'll include it in the next edition. It is available on <AltoDocs>SubSystemsCatalog.Press. A copy can be found beginning on page 37. My thanks to Bruce Nelson of CSL for his efforts in the revision.

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC] and [OLY]<AltoDocs>WawNewsM-YY.press or may be obtained from the editor, Ron Cude, ASD, by messaging <CUDE> or calling Intelnet 8*823-2465.

CeBIT '79: A Report on the Office Equipment Show at the 1979 Hanover Fair

David D. Thornburg and Roy J. Lahr

May 8, 1979

SUMMARY

The office equipment show was larger than last year's show by about 25%. This increase came both in exhibits and attendees. With some expansion planned for next year, this exhibition continues to be one of the most important vehicles for displaying and selling office equipment in the world.

Highlights of this year's fair include the following:

1. In one year Pelikan AG has gone from a pen company to an emergent force in OIS. This came through a series of acquisitions (Lumoprint, Roto), distribution rights (Sanders Technology printer) and internal product developments (including a splendid "electronic secretary", the Variotronic 700). Future acquisitions will include a "major" word processing company (perhaps CPT). This growth is being coordinated by outside consultants.
2. Strong growth is evident in European markets for facsimile, Teletext and personal computers.
3. There is a pronounced trend towards distributed computing and multi-function work stations.
4. Through Mergenthaler and Wang, fair attendees were shown that multiple font documents are both useful and easy to obtain.
5. Technological curiosities included the Iwatsu magnetographic EFE copier/fax transceiver, the Eskofot stream fed copier with image retention, the 3M EFE copier/fax transceiver with CCD input and electrostatic printing, the Olympia ink jet typewriter, and the Siemens ink jet facsimile transceiver.

INTRODUCTION

The annual Hanover fair, located on extensive fairgrounds a few kilometers outside the city, continues to be one of the most impressive exhibitions in the world. For ten days in April, the world is treated to exhibitions of tractors, typewriters, lamps, stereos, telephones, fork lifts, trains, and numerous other articles covering equally numerous industries. Over the last several years, the office equipment portion of the show has grown at the expense of other sections. If one includes telecommunication equipment, the office equipment displays occupy five buildings.

It is impossible for anyone to obtain more than a cursory view of so vast an exhibit. The following report is a distillation of what we found to be new or significant, and thus it generally does not include descriptions of products which were on display in previous years.

This report is divided into several sections including Microfiche, Copiers, Facsimile, Printers, Teletext, Personal Computers and Work Stations with Xerox Products mentioned at the end. The authors are indebted to the Rank Xerox people who provided both a rest station and valuable guidance and especially to Jurgen Wilms and Martin Primke from RX Dusseldorf.

MICROFICHE

This year was not a good one for microfiche products. The splendid portable reader from Fuji Photo Film was removed from the market shortly after last year's fair. Major exhibitors this year included MAP and Krones Transyscorp. The MAP line includes units with automatic random frame access, and the Krones unit projects a fiche image to a rear projection screen on a drafting table. The Xerox 740 was the only new microfiche copier we saw at the fair.

Judging from the paucity of well designed products in this area, microfiche seems to be a field ripe for innovative ideas (assuming that mass memory costs remain high through the mid-80's).

COPIERS

Of the myriad copiers on display, we are choosing to write about only a few machines. More detailed information on copier exhibits is available from the Webster attendees to the Hanover Fair.

Last year two small copiers were previewed which didn't use conventional lens systems. These are now on public display as products available for delivery. The A.B. Dick 980 uses a dual linear array of lenslets and produces fairly crisp copy on plain paper. This machine is made by Agfa Gevert. The second machine is the Minolta EG 301 which uses a "selfoc" lens fiberoptic array in conjunction with a CdS photoreceptor. The copy quality of this machine is quite good, judging from the samples we got at the fair.

Orimig once again featured their forms copier which functions similarly to the Rank Xerox 3600F. One feature of the machine that we didn't see last year is the ability to shift the original relative to a mask to produce a greater variety of output formats.

Eskofot displayed the 1001 plain paper copier with a stream feed system and image retention. Up to 20 copies can be made from a single exposure (with some loss of image quality visible near the end). This copier uses a Se drum (overcoated) and a "selfoc" fiberoptic lens array.

Canon showed two non-products (instead of only one last year). In keeping with "now you see it - now you don't", these were hidden behind moveable panels and shown in conjunction with a magic show whose quality was only slightly higher than that of the copiers. The displayed machines were the Canon color and the NP8500. The color machine was out of tune and the NP8500 looked like a newer version of last year's image retention (NPX) machine. Ordinary mortals were not allowed near the machines.

The Oce exhibit included the Oce 1700 which produces splendid solid area blacks. Any company should be pleased to have that product in their lineup.

The new Mita Copystar (which uses roll paper) has the feature of sensing the edge of the original and automatically adjusting for different length originals.

By far the most bizarre of the copiers at the show was the Iwatsu M/G-6000 magnetographic copier. This machine spends up to a minute scanning in an original and then produces up to 2000 copies of this original at 100 copies per minute. The original is raster scanned (perhaps with a rotating mirror). If needed, digital half toning is performed. The image is then written on a hard metal alloy drum. Once the image is written, the image is developed with a suitable magnetic toner and transferred to paper. While it was suggested that plain paper could be used, the samples we received appear to be on a calendared stock. Since the input is scanned digitally, this "electronic front end" copier can also be used as a facsimile transceiver. The image is rotated 90° between scan-in and print-out.

FACSIMILE

While the Iwatsu copier can function as a facsimile transceiver, the 3M Express 9600 facsimile transceiver can function as a copier (6 copies per minute). This machine uses a CCD input scanner and an electrostatic printer. This machine represents one of the finest entries to date in the Group III facsimile market.

Last year NEC showed a fine (if somewhat large) Group III system. This year they broadened their line with a smaller Group II machine which fits on a table top. This machine uses a CCD input scanner and a thermal printer. The quality of the received image is quite good.

Nexos displayed several machines made by Muirhead. One interesting feature of their system is the ability to take page formatted data from a word processor and convert it into a serial bit map for facsimile transmission.

Siemens showed a machine which uses an ink jet printer similar to that used in their PT-80 typewriter. This machine functioned well. The unit scans 3 lines at a time, scanning in each direction. The printer prints 6 lines at a time, printing in one direction. While increasing the document printing speed, this procedure requires some electronics to sort all the data out.

PRINTERS

Facit had the wide bed Qume printer with two printheads on display. This printer, while not new, was not on display last year. The registration between the two print heads was not as good as one would expect from a daisy wheel printer.

Calcomp was showing the new Gould 5300 wide bed electrostatic plotter which uses 22" paper. The background on the printout was quite bad, but otherwise the image was quite good. One interesting feature is that the Gould machine now loads paper in the same way the Versatec machines do.

Two excellent needle printers were on display. The Printronix P-600 printer functions as a printer and a plotter (using some of the same test patterns used by Versatec) at 600 lines per minute. By using overlapping dots, good solid area blacks are produced. The highest quality needle printer on display was the Sanders Technology Media 12/7 printer distributed by Pelikan. This printer has the ability to place 13 mil dots anywhere on a page to within 1 mil accuracy. Through multiple passes, high resolution characters can be produced which almost rival formed character printing. This printer can handle several fonts at once.

Wang showed their "Intelligent Image Printer," which uses a fiberoptic faceplate CRT in conjunction with a xerographic print engine. Contrary to last year's report, this printer is not the Toshiba machine, but was developed by someone else. The printer operates at 18 copies per minute and can presently handle only two fonts at a time. Wang has 12 fonts available now and will have 120 fonts by 1980.

There were six daisy wheel printers at the fair, including an Olympia correcting backspace typewriter with a five character buffer for automatic character lift-off.

TELETEXT

Several European countries are installing massive timesharing computer systems for the general public to use. These systems communicate over telephone lines to a central computer. Each user has a keyboard and a modem with some local buffering. The home TV is used for the display of up to 24 lines of 40 characters. Services provided by this system are added to the monthly phone bill. Aside from games and home computational tasks, this system can be used to order theater tickets and for electronic mail. Several companies

showed this system at the Fair including SEL/ITT, AEG Telefunken and Siemens. As some indication of the size of the German market for this service, the German Post Office is planning to buy 3,000,000 small printers over the next two years.

PERSONAL COMPUTERS

The personal computer and small business market is opening up in Germany. Last year Commodore was the only major exhibitor of such machines. This year Commodore increased their booth space by a factor of 4 and showed small business configurations with the new 32K byte RAM PET with the good keyboard, plus the dual floppy system and a few printers. Now that Chuck Peddle has rejoined Commodore, it is expected that they will fight to retain their position in the market.

Other personal computer vendors included APF showing the PECOS 1 machine running a superset of JOSS. This machine was not given much attention, probably because it is ugly and because it doesn't use BASIC, as did every other machine at the fair.

Tandy showed an experimental 48K byte TRS-80 in a single housing with built-in display and dual floppies. If this design is marketed, several small business computer companies might find themselves in trouble.

Apple (distributed by Basis) showed their speech recognition and synthesis equipment.

Sharp introduced a new personal computer, the MZ-80K which looks like an old PET (including the bad keyboard design). This Z-80 based machine comes with an extended BASIC and 20 K bytes of RAM (expandable to 48K). The MZ-80K will be introduced in America this October once the floppy disk system is ready.

While priced a bit high for a home computer (\$5,000), the Findex system 100 is a very interesting machine. This self contained computer has a floppy disk, plasma panel display (6 lines x 40 characters), small printer, good keyboard and a Z-80 based processor with 48K bytes of RAM built in. This machine is also available with bubble memory (128K bytes) and a built-in battery pack for true portability. While the internal layout of the machine is messy, the external appearance of this computer is very nice.

WORK STATIONS

Numerous electronic devices are finding their way into the office. As these devices become more sophisticated, functional boundaries disappear. This trend was evident at the fair, with the increased display of multi-function work stations with distributed computing. Since the boundaries between work processing and business computers are becoming more diffuse, we are lumping a variety of tools under the category of work stations. Generally a work station will have a display, keyboard, removable storage media, printer, and a processor to make everything work. Several vendors have done an effective job at minimizing glare from CRT's through the use of a fine mesh screen stretched in front of the display.

Certain of these work stations are primarily oriented towards a single task (e.g., text editing), but increasingly the trend is towards more general functions.

One company which deserves close attention in this regard is Pelikan. This company has been in the pen and pencil business for 140 years and, within the last year, has begun a major move into OIS. This has come about through the acquisition of Lumoprint (copiers) and Roto (printing presses), acquisition of the distribution rights to the Sanders Technology printer, and the soon-to-be announced acquisition of a major WP firm. Internal program developments in conjunction with an outside consultant resulted in the development of the Variotronic 700 "electronic secretary". This machine functions as a calculator, auto dialer, appointment

keeper, clock, etc., and will retail for about \$600. This terminal has a full keyboard, one line bit-map plasma display, communications port, Z-80 processor, 4Kbytes RAM and 2Kbytes ROM in a very nice package. The time from concept to working prototypes was 8 months. This device will be expanded into timesharing terminal soon with little change in price.

Another nicely designed small terminal was displayed by AEG Telefunken. This terminal uses a two line LD display and a full keyboard.

Olivetti continues to produce excellent designs for their calculators and typewriter based text editors. This year the emphasis was towards smaller-lower cost configurations. The Olivetti 221, for example, is a new entry designed to go after the smaller QYX products.

The IBM exhibit contained an impressive array of terminals for myriad applications. Whirlaway was not at the fair (so far as we know).

CPT introduced the CPT 6000, which is a partial page version of the beautiful full page CPT 8000 shown last year. They found that while some customers really wanted full page text display, there were a large number of customers who needed a more cost effective system. The CPT 6000 thus broadens their line without compromising quality. The message seems to be "adapt or die".

The only new Vydec equipment we saw is their laser scanned OCR reader.

The Durango small business computer is marketed under the Video Data name in Europe. This machine has a much nicer industrial design than appeared in previous photographs.

Mergenthaler displayed their Linotext system with the photo printer. This printer handles up to 4 fonts at a time, with each font capable of an arbitrary number of point sizes, since the recording is optical. The data for the printer is formatted and edited on distributed work stations and sent to the printer where it is printed at 30,000 characters/hour. As expected, the linotype font book is a joy to peruse.

A bizarre entry in work stations was provided by Computext with a system for M.D.'s to write their reports on. The doctors can use a microphone for dictation, a standard keyboard for text or a massive back lighted touch panel for canned phrases. This system looks almost impossible to use.

Silver Seiko displayed their version of the IBM correcting typewriter - a very nice machine but not nearly as fancy as that by Olympia which uses a daisy wheel and has a 6 stroke buffer. When you push a correcting backspace on the Olympia machine, it backs up and pulls the character off in one keystroke. Olympia also displayed an ink jet typewriter printer which is not commercially available at this time.

XEROX

Xerox and its companies were well represented at the fair. The Rank Xerox booth was unchanged pretty much from last year. New products included the two new small copiers and a microfiche copier. As in the past, the booth was always jammed with people.

Diablo redecorated their booth very nicely and featured the 1380 wide bed printer along with other Diablo printer products. The Diablo Ranger continues to be shown well by Geveke.

Versatec plotters (both 36" and 11") were displayed by WDV.

The Shugart product line was well displayed by Synelec and Century Data Products (with the new Xerox brochures) were shown by Calcomp. Stielow had two Cheshire label printers on

display and Cheshire had their own booth (obtained at the last minute) in which their binders were on display.

All things considered, the Xerox family was well represented. It might be nice to see if, in the future, a few adjoining booths could be used for Xerox family products, located close to the Rank Xerox booth; or, failing that, a lighted "Xerox Family" guide chart at the main Rank Xerox stand.

REPORT ON THE 1979 INTERNATIONAL SUMMER CONSUMER ELECTRONICS SHOW

David D. Thornburg and Roy J. Lahr

I. INTRODUCTION

The summer CES is the second major consumer electronics show held each year. The exhibition ran from June 3 to 6 and was held primarily in McCormick Place in Chicago with overflow in two adjacent hotels. The spectrum of products displayed at the show were primarily demonstrating current and future consumer products to dealers. Our major reason for attending this show, however, was to see new developments in personal computers, video disks, and new office communication products. The remainder of this report is devoted to a description of the products shown in each of these areas. Many of the products we saw were just introduced, and the authors can be contacted for more details.

II. PERSONAL COMPUTERS

Presumably, this market will settle down at some future date with three or four major vendors controlling the market. At this early stage we are seeing many vendors trying out different price ranges and features. The feeling we have at this time is that it is too early to tell who the long-term survivors in this business will be. Computer systems on display ranged from \$300 to over \$3000, with an equally wide range of features. In the following machine summaries some information may be sketchy since some vendors decided to show equipment at the last instant and had no literature to give out. At this point, equipment is fairly similar, and software and distribution will be the major factors determining market share.

APF

APF is a major calculator manufacturer who introduced a personal computer a year ago called the PECOS 1. This machine failed at the marketplace, probably because it used JOSS, a language the general public did not know anything about, and because of its poor overall design. This year APF introduced "The Imagination Machine" which is a full capability personal computer centered around their MP 1000 video game controller. The combined retail price for the two plug-together units which make up the computer will be \$420 when the product is introduced on the marketplace September 1. This computer can use pre-programmed cartridges or programs written in BASIC which are loaded through the built-in cassette deck. The smooth overall lines of the computer coupled with a nice, tactile keyboard make this a nice looking, easy-to-use computer.

Having learned their lesson on JOSS, APF has provided a fixed-point BASIC with many of the functions associated with Microsoft's BASIC. This machine has FCC approval for connection to the home TV and has the capability to display 8 colors. The basic memory configuration is 10K bytes ROM, 9K bytes RAM, with RAM expandable with an additional 32K bytes.

The display has four formats: text only (32 characters x 16 lines), mixed text and low-resolution graphics (64x32 graphic pixels with 8 colors), high-resolution color graphics (128x192 in 8 colors) and high-resolution monochrome graphics (256x192 pixels). The APF Imagination Machine also has built-in music synthesis. Planned expansion will include additional RAM, 9600 baud serial port, printers, floppy disks, AC controller, and a modem with direct connection to the phone line. At \$420 list for the base machine, APF should do quite well.

Mattel

The Mattel Intellivision is physically similar to the APF system in that it is composed of two combinable units, each of which can be purchased separately. The similarity stops there however, since the Intellivision cannot be programmed by the user. Mattel explained that a market survey had shown that most home computer users do not want to write their own programs, and therefore they decided to not make the machine user programmable. When asked if they ever heard of Videobrain, who pretty much went down the drain after making a similar decision, they were quizzical (video who?).

In any event, Mattel doesn't bother with technical specifications. Their \$500 price tag (based on \$350 dealer cost) will not deter dealers . . . and maybe customers . . . who will be subjected to advertising in which "Mattel Electronics will spend more money than ever before in their history to introduce a new product."

RCA

Some time ago, RCA introduced the VIP single-board computer which was about the size of the KIM-1. Apparently, customer response was sufficient to allow the introduction of the VIP II. This is a "bare bones", skimpily packaged unit that would mainly appeal to the hobbyist seeking absolute lowest cost. The VIP II's major detractor is the lack of a tactile keyboard. The VIP II is available in a 4K byte (\$300) and 8K byte (\$400) RAM versions. Each unit uses the CDP 1802 microprocessor and is expandable to 32K bytes and has 12K bytes of ROM with a resident extended BASIC. The user supplies a monitor (8 colors are available) and a cassette recorder for program storage and loading. The display operates as a bit map (1024 pixels) which only supports 11 lines of 16 characters. The VIP II has a musical tone generator which covers 4 octaves.

Two buffered I/O ports make this a potentially useful controller. If it had a better keyboard and a "general customer" packaging orientation, it would be a much more attractive offering.

Interact Electronics

The Interact Model One computer system is not likely to be a big seller, yet it has many qualities which make it interesting to use. The basic configuration is housed in a nice cabinet with a tactile keyboard (which needs some reduction of key switch closure force plus a two stage "Selectric feel" in our opinion), with a built-in cassette recorder, 16K bytes of RAM, 2K bytes of ROM, two 4-bit input ports and two 6-bit ADC's. This configuration retails for \$500. For \$600 the above unit is provided with two RS-232 ports as well.

This 8080A based machine supports a coarse bit map display (2K bytes) and is soft loaded with either canned programs or one of several BASIC interpreters. The Interact computer supports a color display with multiple windows and produces not only musical tones, but sound effects as well.

Future efforts by Interact will be on a Z-80 based micro-computer (or other better chip set) with a higher resolution (8K byte) bit map display. In the new machine, portions of the display can be turned off if the user needs more memory.

Atari

The Atari 800 and 400 personal computers were on display. These two machines are among the most cleanly designed and function-packed computers to enter the market this year. Both computers interface to either a TV or dedicated monitor, as desired. The Atari 400 (which will retail for \$550) has 8K bytes of RAM and BASIC is provided in a plug-in ROM

cartridge. The user must buy a program recorder (\$90 extra) to store programs. The Atari 400 uses a flat-panel keyboard, but otherwise is nicely packaged.

The Atari 800 comes with 8K bytes of RAM and extended BASIC in ROM. In addition to having a nice keyboard, this computer has a high-speed serial port to access up to 4 floppy disks and a printer. The 800 memory can be expanded internally to 48K bytes. The 8K byte version has a list price of \$1000, which includes the cassette recorder. The disk drives are \$750 each and the printer will retail for \$600.

Both the 800 and 400 have 32 bits of parallel I/O and a special display controller chip for high-resolution color graphics. These computers operate with mixed text and graphics (character generated text) as does the APF machine. Both machines have music capability.

The Atari 800 is among the most versatile of the new machines introduced this year. If it truly will get aggressive marketing by Sears and Wards and perhaps Penneys and others, it could become a top seller.

If this marketing scheme isn't enough, the display format of 40 characters x 25 lines is compatible with the CCITT Teletext standard. Since Atari is owned by Warner Communications, who owns Warner Cable, who owns Qube, who is presently doing bidirectional cable experiments, one can assume that the display format was not chosen at random.

Texas Instruments

The long awaited entry of TI into personal computers was decided at the last possible moment. The TI-99/4 home computer comes with a dedicated Zenith color monitor for a combined price of \$1150. This was the only 16-bit home computer outside of the Mattel Intellivision. The 99/4 comes with Microsoft's BASIC (14K bytes) in ROM. The display operates with mixed text and graphics and uses 16 colors. The display format is 24 lines of 32 characters and musical sound is available in three simultaneous voicings (sounds better than most others).

Since no recorder is provided at that price, the customer must also buy a cassette recorder for program storage.

Contrary to previous rumors, this computer does not yet have PASCAL. It does, however, run the Speak 'N Spell chips. One wonders what all the waiting was for.

The Golden Oldies

With the surge of new names into the field, one wonders what will become of last year's heroes. The remainder of this section will deal with the "old established" personal computer companies.

Exidy is moving towards business systems. The Sorcerer is now being offered with a combination display/dual floppy disk drive with the emphasis on business applications. By the time the whistles and bells are installed, a system price will exceed \$4000. This system is quite powerful, and its ability to handle different fonts (we saw Farsi being typed from right to left, for example) should make it desirable for soft display WP applications.

Compucolor continues to show the Compucolor II, which at \$1495 for the 8K byte RAM version is priced in the Texas Instrument class (the Compucolor II has a built-in disk drive at that price).

Commodore is stressing the version of the PET with good keyboard and new ROM's as a "small business computer". For a 16K byte RAM configuration the price is \$995 plus \$895 for a single floppy drive. One PET was shown with a color display and another was shown driving a speech synthesizer. Neither of these was called a product, however.

Northstar is featuring PASCAL on the Horizon computer. The computer itself (exclusive of extras) retails for \$1900, thus making this more in the small business computer range than a home computer system.

Ohio Scientific was showing a new line of Challenger machines. Since their time spans the range from \$300 to \$3000 computers, they do not have to depend on any one market for success.

The **Apple II** was shown with the new music synthesizer board. This capability makes the Apple the best sounding computer. It will be interesting to see how they respond to Atari and Texas Instruments.

III. VIDEO DISK TECHNOLOGIES

We obtained information on two video disk systems. The first was the Philips/MCA/Magnavox Discovision which was introduced in Atlanta a few months ago and is now going to be introduced in Seattle with a \$695 price for the player. The player styling won this device a well deserved design award. The disk can be loaded by an idiot, and fingerprints had no demonstrable effect on playback quality.

The new entry into this market was JVC. Japan Victor had a prototype of their machine locked in a hotel room. While we couldn't actually get to see the machine, we did get some technical details and a photograph. Unlike the Magnavox unit, the JVC machine uses capacitive pickup. While this is also used by RCA, the JVC disk has no grooves, thus allowing single frame viewing. JVC is quoting \$500 for the player and about \$7 for the disks. From what we can tell, the disks are stamped like phonograph records from a graphite-filled PVC which is electrically conductive (such resins are available from Alloy Polymers for \$0.75/lb in small quantities). Since the disk is already conductive, there are no further processing steps needed. The pickup stylus is a sapphire block with an electrode attached to one face. As the block wears, new electrode is exposed, thus prolonging stylus life considerably.

The stylus actuator has 3 degrees of freedom for fast motions (tracking, jump frame and stylus vertical position). The data track is positioned between two reference tracks. Pit spacing in the disk is about 1.4 microns between tracks, and about the same spacing along the track. The disk rotation speed is 900 rpm. With 54000 tracks per side, this gives a normal playback time of 2 hours per disk.

It will be interesting to see how quickly this product gets to market. The main challenge for JVC would now appear to be program material.

IV. GENERAL

One new technical development is the high performance "metal" tape offered by 3M, TDK, Fuji and others. Retentivity is about three times that of chromium dioxide, and coercivity is about double, with a slightly thinner coating. Essentially the idea is to pack the metal "particles" in so tightly that they resemble a continuous, plated coating. Thus the M-H loop is much, much larger than you can get with oxide, perhaps 7 to 10 db more signal at saturation. Thus for the same recording speed you can markedly extend the frequency range. At 15 kHz, you get at least 4-5 db better performance than "improved chromium dioxide", and 10 db better than standard chromium dioxide. Since you need a lot of power to erase the tape (and furnish the bias to record), this obsoletes every tape deck presently in use . . . and gives the

lagging audio industry something new to sell. And, it is claimed by at least two manufacturers that with metal-capable cassette decks at 1 7/8 ips, you get the performance equivalent to reel-to-reel tape decks running at 7 1/2 ips with twice the track width.

As this show was targeted principally towards showing dealers what will be sold, some of the trends at the show were somewhat submerged by the heavy "hype" used to push audio high fidelity, TV and other entertainment items that were relatively easy to sell (for these items customers seem to be mainly captured by the display techniques, aggressive advertising and competitive pricing). When it comes to the more complex user-device interfaces (home computers, calculators, electronic music devices, etc.) the software available and the means of quickly demonstrating the competence and range of application of the product become dominant. Price is important, as is reputation of the company, but customers are becoming wary that what they buy may become obsolete to them because they won't understand how to use the device fully, or that it will be superseded by next year's model. It was interesting to note that most of the home computer people were pushing their equipment as "games, plus" and that calculator people either pushed "same function, but much smaller" . . . or "same size, but a lot more functions". For example, several vendors showed business card sized calculators less than 1/8 inch thick. Hewlett Packard announced that over 700 preprogrammed "solution pacs" were available for either their model 67 or 97 calculators. HP also passed out copies of an article on "how programmable calculators help kids learn".

Many of the home computer vendors were using software developed by other firms, such as Microsoft. All in all, software is going to be a key in gaining customer acceptance, just as distribution systems are going to be key in solidifying methods of offering more sophisticated equipment to users (either underpinning the "Byte Shops", having company-owned outlets, or offering significant marketing support to mass merchandisers, such as Sears, J. C. Penney, or Wards).

Increase in Communication Capability and Usage

I don't believe there was a word processing system on the floor that did not offer some type of communications facility. The interesting trend, however, is that many vendors are responding to the customer demands that it operate in a background mode and not require operator intervention. Most vendors price these communications options separately. Many combinations of protocols are available with TWX, TTY, 2741 and 3780 among the more popular. In one seminar a speaker asked for a display of hands on how many people used a communicating word processing system. The response was 80% as compared to about 10% last year.

Multi-Function Systems and Multi-Function Users

The introduction of more and more multi-function systems is raising the question of who in the office environments will use these capabilities. Many of the office systems of today are installed in centers or work clusters. It is becoming obvious that operators in a word processing center today do not have a need for many of the 'neat' functions that are being offered including Calendar, Messaging, Math, Electronic Mail, etc.

The value of these functions is recognized but office managers are now encountering the organizational problems of placing these systems in work environment different from the traditional WP environments. Jerry Eisen, Corporate Managing Consultant - Office Automation for ITT felt that the largest productivity gains in their offices in the next three year period would come from multi-function terminals placed with the professional or 'knowledge worker'. He did not feel the impact would hit the executive level until 1983-84. Wang is also directing much of their OIS marketing strategy towards the 'knowledge worker'.

User Programming

Many vendors use the words **user programming**. What they offer may be one or more of the following features.

Keyboard Programming - allows a series of keyboard/menu commands to be activated with a single keystroke

Function Programming - getting the machine to perform a function it could not perform before. This can be done in one of three ways:

1. Machine Language Programming
2. Basic, Agol, Fortran programming
3. Vendor Macro Language programming

Most systems on the market today include keyboard programming, many include a vendor macro language and an increasing number are providing a BASIC type language. The general feeling among customer and consultants is that a good multi-function machine must include a BASIC type language to provide total flexibility. Customers understand that they may have to hire a programmer but they want that option.

BATTLE of the GIANTS

Melody Johnson of Quatum Science presented an interesting matrix of leading vendors and their strengths in the disciplines she perceives as being required in an Office of the Future. It is interesting to note that no vendor is strong in all areas and no vendors are strong in the same sets of areas. She recommended that if a customer felt that eventually they wanted to grow into an Office of the Future, they should make a point to deal only with vendors that had strengths in at least four of the defined disciplines. I have changed her matrix slightly by including electronic printers with phototypesetting. The matrix indicates an involvement in the area, it does not give any indication as to strengths and weaknesses.

OFFICE OF THE FUTURE COMPETITORS

REQUIRED DISCIPLINES

VENDORS

	IBM	XEROX	WANG	EXXON	BURROUGHS
Word Processing	X	X	X	X	X
Data Processing	X	Note 1	X	NO	X
Copier	X	X	NO	NO	NO
Phototypesetter/Printers	X	X	X	NO	Note 2
Voice Dictaphone	X	NO	NO	X	NO
Facsimile	NO	X	NO	X	X
WP/DP	X	Note 1	X	NO	X
Communications Satellite	X	X	NO	NO	NO

Note 1 No, Unless XCS is considered

Note 2 They are offering the B-9270; a Xerox 9700.

There were still a few small companies at the show but basically the field is dominated by large multi-disciplined companies. These have sufficient capital to develop and market the hardware and software applications to meet the evolving needs of the office and take advantage of the ever changing technology. Following is a list of the perceived leaders in the Word Processing industry today and their current product entries. The list is meant to be representative not exhaustive.

VENDOR

PRODUCT ENTRIES IN THE OFFICE MARKET

IBM (OPD,GSD,DPD)

Mag Card/Blind Word Processors
OS/6 Display Records Processor
System 32/34 Small Business Systems
3730 Distributed Processing WP/DP
6670 Document Processor
6640 Ink Jet Printer

XEROX

800 Blind Word Processor
850 Display Typewriter
850 Page Display
VT III Shared Logic System
Diablo Printers
9700 Intelligent Printer
Shugart Media
Xerox 200,400 Facsimile

WANG

Word Processing/Standalone & Shared / Model 5-30
Office Information Systems / Model 125-145
VS Series Small Business
PS Series Personal Computers
CRT, Impact & Line Printers
Graphics Systems Inc. Phototypesetter

EXXON

Vydec 2000 Small Display Word Processor
Vydec 4000 Large Display Word Processor
QYX 5 levels of intelligent Typewriters
QUIP Facsimile Devices

Burroughs

Redactron Word Processing
Context OCR Reader
Graphics Sciences Facsimile Devices

Addressograph Multigraph	AM Word Processors AM/Jacquard WP/DP System AM Phototypesetters ECRM OCR Readers
AES	Lanier Standalones Wordplex Shared Logic and Standalones Lanier Dictation
A. B. Dick	Standalone Word processing Cluster Word Processing Copiers
RAYTHEON	Lexitron Word Processors Raytext WP/DP System
DEC	Word Processing Data Processing
Phillips International	Micom Word Processors Dictation Equipment

FEATURES

The features battle continues but it is amazing to see how similar text processing systems are becoming. One speaker made the observation that one could roll a coin across the exhibit floor and buy the text processor closest to where it stops and not get hurt too badly. NBI and MICOM had the most feature rich system in text processing last year, with excellent statistical typing packages, formula typing and list processing. This year most systems have all of these feature plus more. The battle lines are drawn on yet another set; namely data processing, electronic mail, etc.

TRAINING

There are some interesting training packages showing up.

Wang Laboratories has announced a self-teach package consisting of audio cassettes, workbooks, a quick reference guide and a comprehensive reference manual. This package is complemented by an administrator guide for introducing and monitoring the program. The package is offered in addition to the standard classroom training and sells for \$150.

In addition they are offering a seminar for Office System Supervisors. This seminar includes subjects such as site preparation, hiring and training of personnel, feasibility studies, etc. The cost of this seminar is \$300 plus off-site expenses.

ABDICK offers the following three packages:

- * **Career/Pak** is aimed at lead operators and in-house trainers. It consists of an overview of word processing and its application to the business world. It also includes information on equipment and software, personnel, training and work measurement. This package includes 35mm slides as visual aids.

- * **Skil/Pak** is directed to the operator and consists of a teaching dialogue for whichever ABDICK system is being learned. The text is enhanced by programmed magnetic cards and is coordinated with the training part of the Career/Pak.

- * **Introduction to System Analysis** is aimed at the office manager level and provides an overview of word processing, document distribution, micrographics, etc. This seminar is held off-site.

ABDICK training packages are free of charge.

TECHNOLOGY

Disks - The industry appears to be very interested in the small Winchester type rigid disks recently shown at the NCC. At least two consultants felt that we would see them integrated into word processing equipment by late '79 or early '80. These drives, approximately the same size as a standard floppy drive, contain 20 million characters of storage and cost \$1200-1800. They are being offered by Pertec, Shugart, Memorex and BASF.

Printers - This seems to be an area that is continually changing. More and more vendors are building their own including Wang, QYX and Vydec. Impact, inkjet, dot-matrix and electronic printer technologies are providing users with a broad choice of features.

Impact Impact printers now come in almost as many flavors as one might want. Carriages come in 13 inch (standard), 18 inch, 22 inch and 26 inch widths. Printheads might be selectric ball, plastic or metal daisy wheels or thimble. Both the thimble and daisy may be single or dual headed for multiple fonts or increased speed. The newer printers (Vydec) also are programmed to sense paper position and print wheel.

Ink Jet IBM has two versions of the inkjet printer available at this time. Model I allows printing at 92cps for two paper types plus envelopes. The Model II allows output in three modes; regular 92cps, draft 184cps and production (first copy regular 92cps and subsequent copies draft 184cps). Analysts that panned the printer when it was announced are now rating it not far below correspondence quality. Its multiple font capability and envelope feed have helped in the market acceptance.

Dot-Matrix In the word processing world the Sanders Technology Printer is the major entry in this category. It is a multi-pass matrix impact printer, offering various speeds and print qualities. This system features 15 fonts under program control plus signatures and graphics. Speeds range from 216 cps draft quality to 36 cps correspondence quality. The quality is quite good. It was announced quite a while ago but as of now they are not being offered by any major word processing vendor.

Electronic This area is slowly but surely coming of age. In January, Wang introduced its Intelligent Printer (CRT, 300 dots per inch) and IBM announced the 6670 Document Distribution System (Laser, 240X240). At the NCC last month Canon introduced a 240X240 laser copier that will OEM for approximately \$10,000 and Konishuroku demonstrated a CRT 300 dots per inch printer with an approximate OEM price of \$7,000.

EDUCATION

One of the big jobs ahead of the industry at this time is the need to educate the public in the concepts of word processing and office of the future. This education must be done in order to get funding for college curriculums and attract a work force to the industry. This education will also create a need and a market for our products. The IWP has hired a public relations firm to assist in this and is planning to seek out vendors to underwrite the task.

XEROX ADVERTISING

It is amazing the impact that the Alto/Laurel advertisement, shown on the Hobbit, had within the industry. I fielded numerous questions on the subject and Amy Wohl of DataPro said that close to 50 vendors called her the day after the advertisement was shown asking if Xerox had finally announced. The other impact was that many consultants, previously unable to talk about the product, were able to talk freely. As a result I think we were mentioned in most of the seminars on hardware and software.

EXHIBITS

VYDEC

Vydec was the star of the show with their new softload Series 2000 and 4000. Vydec, a division of Exxon Information Systems (along with Qyx and Quip and many more), announced two new word processing systems, the 2000 and the 4000.

The 4000 has a large (19") green on black display measuring 11" x 14.5". It has the capability to display multiple windows (vertically and horizontally) including two full pages of text side-by-side. It displays multiple fonts (including Greek and Scientific) and multiple pitches (8, 10, 12, and 15) using vector generation technique. Both system feature the new Vydec "natural language keyboard" with the function keys divided into groups of verbs (eg; go to, append to, copy to, adjust) and objects (eg; word, line, sentence, paragraph, page, column, item). Objects can be addressed in an absolute or relative manner; typical instructions might be; *delete paragraph 6, go to paragraph + 2*. Features are quite basic and include automatic storage of each page of text as it is completed, word wrap, global search and replace, document password security, stored formats, double underscore, communications (including fiber optics) packages for math, forms, and sort/select. The impact printer, Vydec made, has automatic paper position sensing and font sensing. There is also an optional wide carriage printer for statistical work. The 4000 series has two models, the 4200 and the 4400. The 4200 has two mini floppy disks which are compatible with the 2000 series. The 4400 uses the regular size (8") floppies and are compatible with Vydec's previous product line. Both models of the 4000, including a standard printer, will sell for \$14,900 and be available in April, 1980.

The 2000 series looks like an oversized 850 line display. It takes two mini floppy disks and can be directly wired to the 4000 via the fiber optics interface. The 2000's display area contains eight lines of text plus four lines of status and commands. The keyboard is a subset of the 4000 keyboard and uses the same natural language commands. Other features of the system are similar to the 4000. This system sells for about \$10,000.

Mid-1980, Vydec plans to announce a series of application packages for the 2000 and 4000 series, including an accounting package with full audit trails. At this time they will upgrade the memory from 64k to 192k.

WANG

Wang is still impressing the industry with their multiple announcements each quarter. The new announcements this quarter shown at the NCC and the IWPA were:

- * Their existing VS minicomputer series (small business) now includes word processing software.
- * A Mailway network which allows documents to be passed among systems with automatic distribution to any name stored in the system directory
- * Their own version of the Qume dual headed printer.
- * 2 new versions of the OIS series the 125 and the 145
- * BASIC on their OIS series for user programming
- * A new series of training packages including seminars for Office Systems Supervisors

Wang is close to offering a single full featured multi-function WP/DP system. They offer so much that their current capabilities and their potential impact on the market should be the subject of a separate memo. I will try to take time to do this in the next months.

LEXITRON

Lexitron previewed RayText, a shared file system resulting from a dual effort with their parent company Raytheon. This combined data and word processing system permits the Lexitron 1303 word processing system to be used as a dumb terminal on a Raytheon Data Systems PTS 1200 system. The 1303's co-exist with ordinary Raytheon terminals and can perform word processing or data processing/data entry via a 3270 emulation. The PTS 1200 supports up to 250 million characters of storage and a variety of printers. The Lexitron terminals are either in the word processing mode or the data processing/data entry mode. When in the word processing mode, however, they do have access to both the large disk storage and printers. With a little more software, this could easily become a fairly good integrated WP/DP system.

AM/JACQUARD

Addressograph Multigraph is continuing to look very good in the market. Their word processing systems include the Amtext 225 and 425. Amtext 225 is a low cost adapter unit that upgrades an existing Selectric to a data entry station. The 425 is a full page text editing station which can be cable connected for communications. They offer the ERCM Optical Character Reader and their full line of photocomposition equipment including their new dry process phototypesetter. Any new user that feels he needs phototypeset output will have to look seriously at this company. Document disks created on their word processing system may be passed directly to the Compset phototypesetting series for final preparation of camera ready copy. This is an advantage over other Phototypesetter/WP interfaces that normally require a 'black box' for conversions.

They also offer the AM/Jacquard series of shared logic and standalone WP/DP systems. Jacquard started on the data processing side of the house and offers data entry packages, accounting, a Basic compiler and other data processing goodies. They have also added an

excellent word processing package and a super user interface. Jacquard also appears to be maintaining its identity; Am/Jacquard will continue to develop and market their systems and AM/Varityper will be responsible for Amtext Word Processors, ERCM Optical Character Readers and Compset Photocomposition equipment.

3M

3M showed their new Series 84 combination word processor/data processor. They offer four levels of software:

1. Basic text editing functions
2. List management
3. Statistical tables plus math
4. Custom Language CL6; a high level data entry language to provide user programming of data processing applications

It is not clear what really can be programmed with this language and what its limitations are. Despite their advertisements this is not what I consider a real merging of the two disciplines. It is also expensive; 18K for a standalone processor.

BURROUGHS

Burroughs Corporation is one of the several firms who have purchased smaller specialized manufacturers to compete in the integrated systems market. Their display, labeled as the Burroughs Office Automation Center, included Redactron word processors, Context Optical Character Readers and Graphic Sciences facsimile equipment. The communicating version of the Redactor series transmits documents directly to and from the digital dex 5100 facsimile unit. The Redactron products were not demonstrating anything new, but they are a good basic word processor with a macro user programming language that has allowed them to stay current in the features race.

SUMMARY

It was an exciting and informative show. I have literature on most of the equipment and will be receiving the proceedings notebook soon. For more details please feel free to contact me.

Inter-Office Memorandum

26

To IFS Users Date July 17, 1979

From Ed Taft and David Boggs Location Palo Alto

Subject How to Use IFS (version 1.21) Organization PARC/CSL

XEROX

Filed on: [Maxc1]<IFS>HowToUse.bravo, .press

Interim File Systems are now in operation at a number of sites. This memo documents the IFS facilities available to most users.

The names of some of the IFSS presently operating are as follows:

<i>Name</i>	<i>Organization</i>
Ivy	Parc (Palo Alto)
Iris	SDD (Palo Alto)
Isis, Sun	SDD (El Segundo)
Ibis	ASD (Palo Alto)
Oly	ASD (El Segundo)
XEOS	EOS (Pasadena)
ADL	ADL (El Segundo)
Erie	WRC (Webster)
Eagle	Corporate headquarters (Stamford)

These names are the ones used to identify specific IFSS to the FTP and Chat subsystems. Information in this memo applies to all IFSS except where otherwise noted.

This edition describes IFS version 1.21. Important changes since the previous version include:

- addition of a mail server and a mail forwarding capability;
- direct transmission of Press files to printing servers;
- addition of a Directory command in the Chat server Executive;
- ability for individual users to 'own' protection groups and change their membership without intervention by an IFS administrator;
- 'Print Directory-Parameters' command renamed to 'Show Directory-Parameters'.

Administration

This section applies specifically to Ivy, the Parc IFS. IFS systems at other sites are administered independently.

We will provide Ivy accounts on request to any member of the Palo Alto community (Parc, SDD, or ASD) who presently has a Maxc account. You must have an Ivy account in order to store files of your own or to access files stored on Ivy by other users.

To obtain an Ivy account, simply send a message to Ed Fiala. You should include in this message the password you would like installed (the password can but need not be the same as your Maxc password, and you are free to change it yourself at any time).

Personal accounts for Parc users will be assigned a disk limit of 1000 pages, and this limit will be enforced strictly at all times. (One IFS page is approximately equivalent to one Maxc page or four Alto pages.) Our intention is not to oversubscribe the actual storage capacity of the file system at any time, thereby avoiding the extreme storage crunches that have been encountered on Maxc.

Allocations for project (file-only) accounts should be negotiated with Ed Fiala. Please limit requests for storage to immediate rather than long-term needs. If a files-only directory is to be maintained by several users, you may request that a user group be assigned; you should designate who is to be permitted to change the membership of the group.

In general, non-Parc users of Ivy will be assigned a disk limit of zero, and no non-Parc files-only directories will be created. Such users are expected to use their own organizations' IFS systems for file storage.

Communication of files among users of different IFSs is a problem, since most people will have accounts on only one server. Our interim solution to this problem is as follows:

1. Commonly-accessed files (e.g., the contents of the <Alto> directory) will be maintained on each IFS. It will be the responsibility of the administrator of each system to keep such files up-to-date, and to move large volumes of data only during off-peak hours.
2. Users outside Palo Alto will be able to obtain accounts on Ivy (with no storage rights) only under special circumstances. We prefer to limit non-Palo Alto usage because access to an IFS via the slow (9600 baud or less) lines can seriously degrade its availability to local users.

Better long-term solutions are presently under development.

How to access IFS

At present, the file services provided by IFS are limited to a fairly basic set. The normal mode of access from Altos is through FTP. The basic operations (Store, Retrieve, List, Delete, and Rename) are invoked through FTP in precisely the same manner as when accessing Maxc. The only difference is that you request FTP to open a connection to some IFS (by specifying its name) rather than Maxc.

You should consult the FTP documentation in the Alto User's Handbook, the Alto Subsystems manual, or [Maxc1]<AltoDocs>FTP.tty, for general information on the use of FTP. IFS can also be reached from Maxc by means of the PUPFTP subsystem.

File naming conventions on IFS are a mixture of Maxc and Alto conventions. The general form of an IFS file name is:

<directory>name!version

All printing characters except '*' are legal in the name. The complete file name may be up to 99 characters long (longer than either Maxc or Alto permit).

All IFS files have version numbers (in the range 1 to 65535) which are defaulted in the usual way, as follows:

Retrieve	highest existing version
Store	next higher version
Delete	lowest existing version
List	all versions

Versions other than the default one may be referred to explicitly (by specifying the version number) or by the notations '!L' (lowest existing version), '!H' (highest existing version), or '!N' (next higher version).

There is presently no facility for automatic deletion of non-current versions, but such a feature may be implemented eventually.

'*' expansion is supported during Retrieve, List, and Delete commands. The expansion is similar to that provided by the Alto Executive; that is, each '*' matches zero or more real characters in a file name.

You may find it convenient to organize your files into *sub-directories* by giving them names such as '<Taft>Memos>HowToUse.Bravo'. Then all files belonging to a particular sub-directory may be accessed by a specification such as '<Taft>Memos>*', and you may direct your attention to a particular sub-directory by establishing a default such as 'Directory Taft>Memos'. The system does not presently attach any important semantic significance to the sub-directory notation, but this may change eventually.

Access via Chat

The current definition of the File Transfer Protocol (the means by which FTP communicates with a file server) limits itself to the basic set of operations mentioned previously. It lacks the means for expressing a number of other essential operations. Improved file access protocols are a topic of current research.

In the meantime, rather than attempting to extend FTP, we have provided an Executive in IFS which you can access by means of Chat (or the bottom *Telnet* window in FTP). This Executive is patterned after the one in Maxc, but has a very limited command repertoire.

Typein and editing conventions are the ones familiar to most users. BS and CTRL-A erase the preceding character, CTRL-W deletes a word, and DEL deletes an entire command or sub-command. Deleted characters are not actually erased from the Alto screen because Chat does not provide such a capability. Most commands must be terminated by RETURN. CTRL-C may be used to abort any command. If you are using any sort of display terminal, timeout will stop at the end of every page (as on Maxc) and IFS will wait for you to type any character before continuing. If you type ahead, this feature is disabled.

The current commands of interest to most users are the following:

@ Login (user) *user-name* (password) *password*

Logs you into IFS. This is necessary before issuing most other commands. Chat may or may not do this for you, depending on the version of Chat you are using.

@ Logout
 @ Quit

Logs you out and closes the connection.

@ Connect (to directory) *directory-name* (password) *password*

Sets your default directory to be *directory-name*, and gives you owner-like access to it. The *password* may be omitted if *directory-name* is your own directory or one to which you have connect privileges.

@ Directory (default) *directory-name*

Sets your default directory to be *directory-name*, but without changing your access rights (and therefore without requiring a password). All subsequent commands dealing with files will behave as if '<*directory-name*>' appeared at the beginning of each file name argument that doesn't name a directory explicitly (i.e., that doesn't begin with '<'). *Directory-name* may include sub-directories (e.g., '<Jones>Memos').

When you issue the 'Directory' command, IFS first displays your current default directory. You may either edit this field (by first backspacing at least one character) or replace it simply by typing the replacement. If you erase the entire field (with CTRL-W), the default directory reverts to your current connected directory.

If the first character of *directory-name* is '>', IFS prefixes the name of your current connected directory. That is, if you are currently connected to directory Jones, the command 'Directory >Memos' is equivalent to the command 'Directory <Jones>Memos'. Also, the outermost '<' and '>' are optional. Note that the foregoing descriptions also apply to the Directory command in the FTP server.

@ DskStat

Prints the number of used pages and the maximum allowed in the connected directory, followed by the number of free pages in the system. One IFS page is 1024 words or 2048 characters, which is equivalent to four Alto pages or approximately one Maxc page.

@ List (files) *file-designators*

Lists the names of all files matching *file-designators*, which is a list of up to 10 file names (separated by spaces), any of which may contain '*'s to denote multiple files. The files matching each *file-designator* are listed in alphabetical order on the basis of the entire file name (including directories and sub-directories, if any). To save space, directory and sub-directory names are printed only when they change, above the list of files to which they apply.

If you terminate the last *file-designator* with a comma followed by RETURN (rather than just RETURN), IFS enters a sub-command mode in which you may specify additional information to be printed about each file:

@@ Type	file type and byte size
@@ Size	size in pages
@@ Length	length in bytes
@@ Creation	date of file creation
@@ Write	date of last write
@@ Read	date of last read
@@ Backup	date of last backup
@@ Times	times as well as dates

@@ Author	creator of file
@@ Protection	file protection
@@ Verbose	same as Type Size Write Read Author
@@ Everything	

Sub-command mode is terminated when you type just RETURN in response to the '@@' prompt. The columns of printout will be aligned properly only if you are running Chat with a fixed-pitch font such as Gacha12 or Gacha10.

@ Delete *file-designator*

Deletes all files matching *file-designators*, which is a list of up to 10 file names (separated by spaces), any of which may contain '*'s to denote multiple files. The version number defaults to the lowest existing version; to delete all versions, you must end each *file-designator* with '!*'. IFS prints out each file name, followed by '[Confirm]'. You should respond with 'Y' or RETURN to delete the file, or with 'N' or DEL to leave it alone.

If you terminate the last *file-designator* with a comma followed by RETURN, IFS enters a sub-command mode in which you may request the following additional actions:

@@ Confirm (all deletes automatically)

IFS will not ask you to confirm deleting each file but will just go ahead and do it.

@@ Keep (# of versions) *number*

IFS will retain the *number* most recent versions of each file and delete all remaining versions. That is, to delete all but the most recent version of each file, specify 'Keep 1'.

On IFS (unlike Maxc), files are deleted immediately; there is no Undelete command. To delete a file, you must have write access to it.

@ Rename *existing-filename* (to be) *new-filename*

Changes the name of *existing-filename* to be *new-filename*. It is permissible to change any part of the file name, so it is possible to move a file from one directory or subdirectory to another by renaming it. The Rename operation requires that you have write access to the file and create access to the directory into which the file is being renamed.

It is permissible to rename a file to itself in order to change its capitalization. Note that a new version of a file always inherits the capitalization of the previous version; renaming a file to itself (i.e., with the same version number) is the only way to defeat this.

@ Print (files) *file-designator*

@ Press (files) *file-designator*

Requests that all Press files matching *file-designator* be sent to your default printing server ('Print' and 'Press' are synonyms). *File-designator* is a list of up to 10 file names (separated by spaces), any of which may contain '*'s to denote multiple files. IFS prints out the name of each file followed by '[Confirm]'; you should respond with 'Y' or RETURN to print the file, or with 'N' or DEL to skip over it.

If you terminate the last *file-designator* with a comma followed by RETURN, IFS enters a sub-command mode in which you may specify the following parameters:

@@ Copies number

Specifies the number of copies of each Press document to print.

@@ Host *host-name*

Specifies the name of the printing server to which the Press files are to be transmitted. This may be either a registered name or an inter-network address of the form '*net#host#*' (don't leave off the trailing '#').

You terminate sub-command mode by typing RETURN in response to the '@@' prompt. In the absence of any sub-commands, IFS will cause one copy of each Press file to be printed on your default printing server. You may establish or change your default printing server by means of a sub-command of the 'Change Directory-Parameters' command, as follows:

@ Change Directory-Parameters (of directory) *directory*
@@ Printing-Server *host-name*

where *directory* is the name of your directory, i.e., your user name. If you have not established your default printing server, IFS will require you to issue a 'Host' sub-command every time you request printing.

Actual transmission of the Press files to the printing server is performed by a background process, so you need not remain connected to IFS while the printing is taking place. If the printing server is down at the time, IFS will queue the files for later delivery. If the Press files cannot be delivered within eight hours, however, the printing request is discarded.

There are presently no facilities for retracting printing requests or interrogating their status. Also note that *only* Press-format files can be printed; IFS checks that every file is a Press file and will refuse to print any file that is not.

@ Change Password (of directory) *directory-name* (old password) *password* (new password) *password*

Changes the password of the specified directory, which must be either your own or the one to which you are presently connected. (Contrary to normal practice, the new password does print out as you type it; this is so that if you make a typing mistake you will be able to see it.)

@ Change Protection
@ Change Directory-Parameters
@ Show Directory-Parameters
@ Change Group-Membership
@ Show Group-Membership

See the section describing protections (below).

@ Sstat

Shows who is presently using IFS, what service they are accessing (FTP, Telnet, or Mail), and the name or inter-network address of the machine they are coming from.

@ DayTime

Displays the current date and time.

@ Statistics

Prints out various operating statistics that are generally of interest only to IFS administrators.

Protections

IFS has a reasonably flexible file protection mechanism, but with a somewhat primitive user interface at present. Fortunately, the default protections are the ones appropriate for most users, so you will probably not need to deal explicitly with protections very often.

Your access to files and directories is permitted or denied on the basis of your membership in *user groups*. Every user is a member of a user group called 'World'. You are a member of another user group called 'Owner' with respect to files in your own directory, and temporarily to files in any other directory to which you connect (using the Connect command in FTP or Chat). Additionally, you may be a member of one or more other user groups with numbers in the range 0 to 61. Such numbered user groups generally correspond to specific projects, and are assigned independently within each IFS by that IFS's administration.

A *file protection* specifies, for each individual file, what types of access are permitted to which groups. There are three types of file access: *read*, *write*, and *append*. If you have read access to a file, you are permitted to read (i.e., retrieve) its contents. Similarly, write access permits you to overwrite, delete, or rename the file, and append access permits you to append to an existing file, even if you don't have write access. IFS does not yet provide facilities for appending to files, but such a capability may be implemented in the future.

The standard default file protection permits read, write, and append access to the Owner and read access to the World. Hence if the file is in your own directory or the directory to which you are connected, you may do anything to it; otherwise you may only read it. But, for example, if the file protection also permits write access by group 3, and you are a member of group 3, then you may overwrite (or delete or rename) the file, even if it is not in your directory or the directory to which you are connected. Note that the read, write, and append access types are independent. It is therefore possible, though perhaps not particularly useful, for a file protection to permit writing but prohibit reading by some user group.

In addition to the protection associated with each file, there are some protections associated with a directory as a whole. The first is the *default file protection* for files in that directory. When a file is created, its protection is assigned in one of two ways. If there is an existing version of the same file, then the new file inherits its protection. More precisely, when version *n* of a file is created, it inherits the protection of the highest-numbered existing version less than *n*, if there is one. Otherwise, the protection assigned is the default file protection of the directory in which the file is being created.

There are two additional types of access to the directory: *create* and *connect*. If you have create access to a directory, then you are permitted to create new files in that directory. If you have connect access to a directory, you are permitted to connect to that directory without giving its password. As with file protections, these types of access are granted or denied individually to Owner, World, and each numbered user group. The standard directory protection permits create and connect access only to the owner.

The Chat Executive contains several commands by means of which you may manipulate protections of files and directories.

@ Change Protection (of files) *file-designators*
@@ *sub-commands*

Changes the protection of all files matching *file-designators*, which is a list of up to 10 file names (separated by spaces), any of which may contain '*'s to denote multiple files. You specify the changes to be made by means of one or more of the following sub-commands:

@@ Read (access permitted to) *groups*
 @@ Write (access permitted to) *groups*
 @@ Append (access permitted to) *groups*

where *groups* is a list of up to 10 instances of 'Owner', 'World', or group numbers (separated by spaces) to which the specific access type is to be granted. 'None' may be used in place of *groups* to specify that access is to be denied to all groups. You may precede a sub-command by the word 'No' to specify individual groups to which access is to be denied. The changes take effect when you type RETURN immediately after the '@@' prompt.

Normally, the changes that you specify by means of these sub-commands are *incremental*. That is, the only access/group combinations that are changed are the ones you mention explicitly, while all the remaining ones are unchanged. However, there is an additional sub-command,

@@ Reset (all existing access)

that denies all types of access to all groups. In this case, the entire file protection is changed to permit only those access/group combinations that you enable explicitly.

You may change the protection of any file to which you presently have write access, and of any file in your own directory or one to which you are connected regardless of its protection. That is, you can change the protection of any file of your own even if its present protection does not permit read, write, or append access by you.

@ List ...

The 'Protection' sub-command to the 'List' command (described previously) displays a file's protection thus:

R: *groups*; W: *groups*; A: *groups*

For example:

R: Owner World; W: Owner 3 19; A: None

@ Change Directory-Parameters (of directory) *directory-name*
@@ *sub-commands*

Changes the information associated with the directory as a whole in the manner specified by the *sub-commands*. The directory must be either your own or one to which you are connected.

You may change the default file protection by means of the 'Read', 'Write', and 'Append' sub-commands in the same manner as in the 'Change Protection' command. Additionally, you may change the create and connect access using the sub-commands:

@@ Create (access permitted to) *groups*
 @@ Connect (access permitted to) *groups*

The 'No' prefix may be applied to these as well as to the others.

The 'Reset' sub-command requires an additional keyword to specify what it is that you wish to reset:

```
@@ Reset Default-File-Protection
@@ Reset Create-Protection
@@ Reset Connect-Protection
```

You may change your default printing server by means of the sub-command:

```
@@ Printing-Server host-name
```

The changes are not actually made until you type the confirming RETURN in response to the '@@' prompt.

@ Show Directory-Parameters (of directory) *directory-name*

Displays all information about *directory-name*, and additionally prints some other parameters, such as the disk limit, that may be changed only by an IFS administrator. If *directory-name* is your own directory, your user group membership is also shown.

An IFS administrator can change any directory parameters for any user. Additionally, an administrator can assign you to be the *owner* of one or more user groups. If you are the owner of a group, you are permitted to change and examine the membership of that group, using the following commands:

```
@ Change Group-Membership (of group) group
@@ sub-commands
```

The sub-commands are one or more of the following:

```
@@ Add user-name
@@ Remove user-name
```

These cause the specified users to be added to or removed from the group. The sub-commands take effect immediately. You exit sub-command mode by typing RETURN immediately after the '@@' sub-command prompt.

@ Show Group-Membership (of group) *group*

Displays the list of users who are members of the specified group. This command takes a long time to complete, because it has to read the directory parameters of every user in the system.

Mail server

Note: at the time this memo was written, the facilities described in this section had not yet been put into operation. You should await an announcement from your local support organization before attempting to use these facilities.

IFS optionally makes available a mail server compatible with the Laurel message system. Each geographical area has a registry of mailboxes for all Alto users in that area; at present, the registries are called PA (Palo Alto), ES (El Segundo), XEOS (Pasadena), and WRC (Webster). In the current implementation, each registry corresponds to a single mail server machine that contains all the mailboxes within that registry; that is, the registry names are simply aliases for machines. The PA registry is on Maxc1 and the other registries are on local IFSs.

If you are in Palo Alto, you will be assigned a mailbox in the PA registry (i.e., you will be given an account on Maxc1); if you are outside Palo Alto, you will be assigned a mailbox in your own local registry. In any event, your registry must be identified in your Laurel.profile, which should look something like this:

Registry: *registry-name*
Hardcopy: *printer-host-name*
Printed-by: \$

To send a message to a user whose mailbox is within your own registry, you need only specify that user's name when you are composing the recipient list in Laurel. However, to send to a user in some registry other than your own, you must specify a recipient name in the complete form

user . registry

For example, if your own registry is ES (El Segundo) and you wish to send a message to Jones, who is also in El Segundo, you need only specify 'Jones' (though it is also correct to say 'Jones.ES'). But if you wish to send a message to Smith in Palo Alto, you must specify 'Smith.PA'.

File backup

Reliability of file storage is accomplished by two facilities, both of which are now operational. First, we have a Scavenger capable of reconstructing the IFS directory from redundant information kept in the file system. We expect to be able to recover from most file system crashes in this manner, with no loss of user files.

Second, we have an automatic backup system that periodically copies files to a backup disk pack. The backup system runs between 2:00 and 5:00 a.m. every day (users accessing IFS during that time may notice some significant degradation in performance). During each backup run, all files not previously backed up or last backed up more than 30 days ago are copied.

This backup system serves two purposes. First, if the file system fails catastrophically in a way that the Scavenger can't recover from, we will be able to reconstruct the file system from backup, with at most one day's files lost. Second, files accidentally deleted or overwritten by users will usually be recoverable if the loss is noticed within 30 days. (The recovery procedure is not particularly convenient, so please don't depend on it as a regular service.)

Present limitations and future plans

IFS now provides facilities sufficient to make it a useful service. We are presently considering how much additional effort to invest in IFS development. This topic is covered in some detail in a separate memo, available as [Maxc1]<IFS>StatusAndPlans.bravo and .press.

A major concern is that of performance of the file system. There is insufficient capacity (particularly main memory) in the IFS Alto to support more than a small number of simultaneous users. While we expect eventually to effect some improvements by better implementation and fine-tuning, there is little prospect for really major performance gains.

We are presently imposing a limit of five concurrent connections (FTP and Chat users combined), at which point the system will refuse to accept additional service requests. To prevent idle users from tying up these precious slots, the IFS will break connections after a relatively brief period of inactivity. (The limited number of concurrent servers is one of the reasons we wish to discourage large-scale file transfer activity on the Parc IFS by users

outside Palo Alto, since the low speed of the communication links causes such transfers to take a long time to complete, thereby tying up servers.)

We would be pleased to receive reasonable suggestions for changes or improvements in the set of facilities provided by IFS. However, please be conscious of the limited manpower available for implementing such improvements.

Acknowledgments

Implementation of IFS would have been impossible without the assistance and cooperation of several individuals who have contributed considerable effort in support of this project. Peter Deutsch provided the Overlay, VMEM, and ISF packages and implemented a number of improvements needed by IFS. Ed McCreight made available his B-Tree package, which is used for maintaining user directories, and likewise contributed IFS-related improvements. Bob Sproull and Roger Bates sank considerable energy into the Trident disk hardware, microcode, and software to make it work reliably. And Steve Butterfield implemented all the Mail facilities and made several other improvements.

ALTO SUBSYSTEMS CATALOG

37

July 1979

Filed on [Maxc]<AltoDocs>SubSystemsCatalog.Press

This catalog lists and briefly describes the various Alto subsystems. A subsystem is defined to be any program that runs on the Alto whether under control of the standard executive or as a stand-alone. Each subsystem has an entry in this form:

PROGRAMNAME: Description
DOCUMENTATION: FilePath

To simplify locating a piece of software to perform a specific function, a functional cross reference of Alto subsystems is provided beginning on page 9. New subsystems are continually being developed throughout the Whole Alto World. This catalog is maintained and distributed by the *Alto User Support Coordinator* (Ron Cude). If you have or know of a subsystem which you feel should be included, please provide the above information to the *Coordinator*. The following criteria should be met before a subsystem is cataloged and distributed:

1. The program should be in general use at the installation.
2. Support should be provided to at least fix major bugs.
3. Satisfactory documentation must be available and up to date.
4. Management must indicate the sensitivity of the item to outside disclosure.

The subsystems listed are generally available from your local File Server or Maxc under the <ALTO> directory; the supporting documentation is found in the <ALTODOCS> directory unless otherwise noted. If you do not have access to a File Server or Maxc, contact the *Coordinator*.

AIS: A driver subsystem which interacts with the user to perform a set of standard operations on imaginal data stored as AIS (Array of Intensity Samples) files.
DOCUMENTATION: AIS-Manual.Press.

AISDUMP: A part of the AIS system to write out to the Diablo disk the decimal values for pixels within a specified window.
DOCUMENTATION: None.

AISMAGNIFY: A part of the AIS system to magnify or minify AIS format images in either 1 or 8 bit/pixel form.
DOCUMENTATION: [Erie]<AIS>Memos>AISmagnify.Press.

AISSHOW: A part of the AIS system that displays an AIS image file on the Alto display.
DOCUMENTATION: [Erie]<AIS>Memos>AISshow.Press.

ANALYZE: A part of the Design Automation System that transforms logic diagrams produced using SIL into a file which can be input to the GOBBLE wirelister.
DOCUMENTATION: [Maxc]<SIL>SilManual.Press.

APROM: Superseded by PROM.

ASM: An assembler for the Alto machine language which produces relocatable files compatible with the BCPL loader, BLDR.
DOCUMENTATION: ASM.tty or Subsystems.Press.

BCA: A basic cross-assembler for micro-computers.
DOCUMENTATION: BCA.Press.

BCPL: A compiler for Alto BCPL language which produces files relocatable with the BLDR loader.

DOCUMENTATION: BCPL.Tty/.Press.

BLDR: A loader for the relocatable files produced by BCPL and ASM.

DOCUMENTATION: BCPL.Tty/.Press.

BRAVO: A text editor having extensive formatting and hardcopy facilities.

DOCUMENTATION: ALTO User's Handbook, BRAVO Course Outline, BRAVO.Press and BRAVOSUMMARY.Press.

BREETEST: A B-Tree dictionary maintenance program used to support the PROOFREADER data base.

DOCUMENTATION: ProofReader.Tty.

BUILD: A part of the Design Automation System that helps with the data management aspects of building boards and keeping the design automation data files current.

DOCUMENTATION: [Maxc]<SIL>SilManual.Press.

BUILDBOOT: A program for constructing type B bootfiles from either an executable (BLDR out) file or a segment file.

DOCUMENTATION: BuildBoot.Tty or Subsystems.Press.

CALCULATOR: A bootfile that pictures a TI SR-52 on the display which is operated by using the mouse to select the appropriate keys. It is not programmable.

DOCUMENTATION: SR-52 Manual.

CALLFTP: A subset of FTP which always lives in the operating system. CALLFTP is much smaller than FTP and can be used when space is tight.

DOCUMENTATION: FTP manual.

CHAT: A program for establishing PUP Telnet connections between a pair of cooperating parties. Its chief function is to permit Alto users to talk to Maxc.

DOCUMENTATION: ALTO User's Handbook, CHAT.tty or Subsystems.Press.

CLEANDIR: A program to garbage collect a disk file directory (but not disk space).

DOCUMENTATION: Subsystems.Press.

CONDENSE: A program to retrieve the screen bitmap from the SWAT and SWATEE files for display or output to an AIS or Press format disk file.

DOCUMENTATION: Menu is self-explanatory.

COPYDISK: A program for copying entire diskpacks. It will copy from one drive to another on the same machine, or between drives on separate machines via a network using Diablo Model 31/44 and Trident T-80/T-300 disks.

DOCUMENTATION: CopyDisk.Tty or Subsystems.Press.

COPYFROMDRIVE1: A program to copy an individual file from DP1 to DP0 of a dual disk Alto.

DOCUMENTATION: Subsystems.Press.

CREATEFILE: A program to create a file of a given size, attempting to allocate it on consecutive disk pages.

DOCUMENTATION: Subsystems.Press.

CRTTEST: A diagnostic program used to adjust the Alto display linearity. Three different sized grids are displayed in rotation (press any key to change grids).

DOCUMENTATION: none.

CRUMPLE: A program to compress and, optionally, encrypt data files. The resulting files can be stored or transmitted but must be expanded and decrypted before processing by Alto programs.

DOCUMENTATION: Crumple.Press.

DDS: A program to manage an Alto diskpack. Facilities are provided to display filenames, lengths, creation-read-write dates, and contents, internal operations such as delete, and rename, and external operations such as Send and Execute.

DOCUMENTATION: ALTO User's Handbook or DDS.Tty or Subsystems.Press.

DIEX: A Diablo disk exerciser similar in operation to TRIEX.

DOCUMENTATION: Internal to program.

DMT.BOOT: A memory diagnostic and statistics gathering program.

DOCUMENTATION: DMT.Tty or Subsystems.Press.

DO: A wonderful program giving a parameterized interface to the executive. More powerful than CM files, and useful with the IF program.

DOCUMENTATION: DO.Press.

DPRINT: A program to type text files on the Diablo HyType printer.

DOCUMENTATION: DPrint.Tty or Subsystems.Press.

DRAW: An interactive illustrator program for creating black-and-white or color pictures composed of lines, curves, and text captions. The illustrations can be output to a one page press file.

DOCUMENTATION: ALTO User's Handbook plus DrawNews.Press, an on-line manual (part of the DRAW package), and Draw-Summary.Press.

EDP: An Ethernet interface diagnostic.

DOCUMENTATION: EDP.Press or EDP.Bravo.

EFTP: A small but inefficient Ethernet FTP protocol.

DOCUMENTATION: EFTPPackage.Tty.

EMPRESS: A program to send press and text, e.g. bravo format, files to a press printing server. Simple formatting options such as Tab and FormFeed are available.

DOCUMENTATION: EmPress.Tty or Subsystems.Press.

ERP: A program that listens for event packets on the Ether. Useful for monitoring and gathering statistics.

DOCUMENTATION: ERP.Press.

EXECUTIVE: The Alto command processing subsystem, the intermediary by which users generally invoke other subsystems and perform several operations on the Alto file system. Normally invoked by the boot operation.

DOCUMENTATION: Executive.Tty or Subsystems.Press.

FIND: A subsystem to search one or more text files for a user supplied string at very high speed and then display each line containing an occurrence of the pattern on request.

DOCUMENTATION: Find.Tty or Subsystems.Press.

FRED: A part of the Font Creation System, it is used to create and/or edit "splines" (i.e. outlines) of characters.

DOCUMENTATION: [Maxc]<KGR-DOCS>Fred.ears.

FTP: A file transfer program to store and retrieve files between an Alto and another Alto, Maxc, or File Server. It also supports a Telnet connection that is similar to CHAT in purpose and operation.

DOCUMENTATION: FTP.Tty or Subsystems.Press.

GOBBLE: A part of the Design Automation System that generates a wirelist and routing information for a single board given one or more node list files generated by ANALYZE.

DOCUMENTATION: [Maxc]<KSIL>SilManual.Press.

GYPSY: A modeless text editor using both keyset and mouse, and having a "filing cabinet" interface which provides some file management facilities beyond the normal Alto filing system. Used in applications where limited formatting facilities are required such as programming.

DOCUMENTATION: Under development.

HARDCOPY: A wonderful program that retrieves and prints files from anywhere, your Alto, Maxc, a File Server, etc.

DOCUMENTATION: HardCopy.Tty.

ICARUS2: A part of the ICARUS2 System, it is an interactive program for actually laying out printed circuits and manipulating the resulting files.

DOCUMENTATION: [Maxc]<ICARUS>Icarus2doc.Press, ICtools.Press.

IF: A program allowing conditional execution of executive commands. Useful in conjunction with DO files and CM files.

DOCUMENTATION: IF.Press.

IFD2: A part of the ICARUS2 System that turns an ICARUS2 file into human readable form, describing each symbol and its contents.

DOCUMENTATION: [Maxc]<ICARUS>IFD2doc.Press.

IFS: The Interim File System server that provides one end of the file transfer facility and maintains the files and directories on Trident T-80/T-300 disks.

DOCUMENTATION: [Maxc]<IFS>IFSdocuments.Press.

IFSCAVENGER: A subsystem to check and correct Trident T-80/T-300 diskpacks from File Servers or other Trident File Systems.

DOCUMENTATION: [Maxc]<IFS>ScavOp.Press.

INSTALLSWAT: An installation program to install the SWAT debugging system on your disk.

DOCUMENTATION: None.

KAL: A kaleidoscope program.

DOCUMENTATION: None.

KEYTEST: A diagnostic program that displays the Alto keyboard, keyset and mouse. The depressing of any key(s) is reflected by inverting (white to black) the display of that key on the screen. If the keyboard displayed doesn't match the one you are using, move the cursor to the bottom of the display and hit any mouse button.

DOCUMENTATION: None.

LAUREL: A Maxc MSG compatible, display-based, message system that runs on your Alto.

DOCUMENTATION: [Maxc]<Laurel>Laurel.Press and a system tutorial retrieved with Laurel software.

LISTSYMS: A programming aid to convert a .Syms file (produced by BLDR) to a useful, human readable form.

DOCUMENTATION: ListSyms.Tty or Subsystems.Press.

LOGICPROM: Superseded by PROM.

MADTEST: A diagnostic that runs tests on an Alto's RAM, ALU, and emulator.

DOCUMENTATION: Alto User's Primer.

MAILCHECK: A simple subsystem that checks for mail at some other host (e.g. Maxc) via the Ethernet.

DOCUMENTATION: MailCheck.Tty or Subsystems.Press.

MARKUP: An illustrator used to add pictures consisting of lines, areas, mouse tracks and text captions to formatted documents, i.e. Press files. It may also be used to simply display press files.

DOCUMENTATION: Alto User's Handbook.

MENUEEDIT: A program that edits menus for the BCPL menu package.

DOCUMENTATION: Menu.Press.

MIKE: A part of the ICARUS2 System that transforms ICARUS2 files into a form suitable for the Mann 3000 pattern generator.

DOCUMENTATION: [Maxc]<ICARUS>MikeUserDoc.Press, MikeDoc.Press.

MOVETOKEYS: Obsolete, non-functional.

MU: The Alto Microcode assembler.

DOCUMENTATION: MU.Tty or Subsystems.press.

NEPTUNE: A small, fast DDS-like program for manipulating your Alto directory.

DOCUMENTATION: Neptune.Press.

NETWORK EXECUTIVE: The executive obtained by booting from the Gateway over the Ethernet that provides a convenient way to call "bootfiles" such as FTP or CopyDisk.

DOCUMENTATION: NetExec.Tty.

NPGR: Obsolete part of the Sil system.

NPPR: Obsolete part of the Sil system.

OEDIT: A subsystem for displaying and modifying Alto files in octal. Up to four files may be simultaneously viewed while one of them may be modified.

DOCUMENTATION: Oedit.Tty or Subsystems.Press.

ORBITTEST: The ORBIT interface diagnostic.
DOCUMENTATION: [Ivy]<Spruce>ORBITtest.Press.

PACKMU: A program to convert the output of MU (an MB file) to a "packed RAM image" which is easy to load into the RAM using RPRAM.
DOCUMENTATION: PackMU.Tty or Subsystems.Press.

PEEK: A program which listens to the Ethernet for PeekReports and EventReports. It can also serve as a bootserver and Ethernet Echo server for use with EDP.
DOCUMENTATION: DMT.Tty.

PEEKPUP: A small subsystem enabling one to peek at Pups going to and from a particular Ethernet host; a debugging aid for new Pup software.
DOCUMENTATION: PeekPup.Tty or Subsystems.Press.

PEEKSUM: A subsystem that summarizes the error reports sent to PEEK by DMT.
DOCUMENTATION: DMT.Tty.

PNEW: A subsystem to edit PROM files, drive the *Pro Log* PROM blower and verify previously programmed PROM's.
DOCUMENTATION: PromManual.Bravo

PREPRESS: A part of the Font Creation System that takes "spline character definitions", usually created by FRED, and generates scan-converted characters, spline and character dictionaries, readable listings describing the dictionary's content, and a "widths" file for use by text formatting programs.
DOCUMENTATION: [Maxc]<GR-DOCS>PrePress.Press.

PRESS: A subsystem to print full press files on press printers.
DOCUMENTATION: [Maxc]<GR-DOCS>PressOps.Press.

PRESSEEDIT: A program to combine Press files together, convert Ears files (generated by Pub and Bravo) to Press format, selecting certain pages from a Press or Ears file, or to add extra fonts. The output is a Press file.
DOCUMENTATION: PressEdit.Tty or Subsystems.Press.

PROM: A subsystem to edit microcode, drive the Alto PROM blower and verify PROMs.
DOCUMENTATION: Prom.Bravo.

PROMDIAG: Superseded by PROM and PNEW.

PROOFREADER: An interim English text proofreader that produces an output file listing the questionably-spelled words.
DOCUMENTATION: ProofReader.Tty.

PUPTEST: A PUP protocol and network integrity test program.
DOCUMENTATION: None.

PUT: A program for transferring files between the disks of a dual-drive Alto. Its function is also performed by the more comprehensive NEPTUNE.
DOCUMENTATION: Internal to the program.

QED: An in-core line editor used primarily for programming. The file is limited to about 1500 lines of BCPL.

DOCUMENTATION: QED.Tty or Subsystems.Press.

RAMLOAD: A microcode loader that uses the output of the microcode assembler, MU. Also contains a limited amount of CRAM test and verification functions.

DOCUMENTATION: RamLoad.Tty or Subsystems.Press.

RAMTIMING: A bootfile diagnostic program to test the Alto CRAM. Its function is also performed by the more comprehensive MadTest.

DOCUMENTATION: None.

READPRESS: Reads Press files and displays a text-listing of the entity commands, DL strings, etc.

DOCUMENTATION: Subsystems.Press.

RENAME: Obsolete replacement for the Executive's Rename command.

RPRAM: A microcode loader that loads a packed RAM image (generated by PACKMU) into the CRAM after checking the constant memory.

DOCUMENTATION: PackMU.Tty or Subsystems.Tty.

SCAVENGER: A subsystem for checking and correcting Alto disk packs.

DOCUMENTATION: Scavenger.Tty or Subsystems.Press.

SETTIME: Obsolete.

SHOWAIS: A bootfile that halftones and displays 8 bit/pixel AIS files stored on a remote file server. (Use of this program significantly loads the file server, limiting overall performance. Do not use it frivolously.)

DOCUMENTATION: [Ivy]<Maleson>ShowAIS.Bravo.

SIGMA: A subsystem to transfer arbitrary files between an Alto and a SIGMA 3 over the Ethernet.

DOCUMENTATION: "Ethernet Software for Data Transfer between the SIGMA 3 and an ALTO", a Xerox Internal Report, Accession No. X7704459.

SIL: A part of the Design Automation System, it is an illustrator for the creation of logic and simple line diagrams. The output can generate Press files or be processed by ANALYZE for circuit design.

DOCUMENTATION: [Maxc]<SIL>SilManual.Press.

SORT: A very small subsystem which will sort files containing less than 1000 entries delimited by a carriage return.

DOCUMENTATION: Subsystems.Press.

SPRUCE: A printer server that utilizes the ORBIT buffer to drive Press printers, e.g. Dover and Sequoia.

DOCUMENTATION: [Maxc]<Spruce>SpruceManual.Press.

SWAT: A emulator-level code debugger with BCPL oriented features used with the Alto operating system.

DOCUMENTATION: Swat.Tty or Subsystems.Press.

SYS.BOOT: The operating system boot file on the Alto disk.
DOCUMENTATION: Subsystems.Press.

TFU: A file utility used to initialize a Trident pack with a virgin file system and to perform various file copying, deleting, directory listing operations. This is not a part of the Interim File Server System, rather it initializes and maintains packs operated on by the TFS package.
DOCUMENTATION: TFS.Tty or Subsystems.Press.

TRANSFILE: A part of the ICARUS2 System that translates the intermediate files generated by Mike into human-readable form. The files it produces are fully instantiated.
DOCUMENTATION: [Maxc]<ICARUS>TransfileDoc.Press.

TRIEX: A Trident diagnostic used to debug and exercise Trident disk drives.
DOCUMENTATION: Self-contained.

TYPE: A functional replacement to the Executive supplied "type.~" that displays a larger page, suppresses Bravo trailer information, can skip forward and backward, etc.
DOCUMENTATION: Type.Tty.

UGH: An in-core text editor utilizing both mouse and keyset. While some formatting facilities are available, it is used primarily for programming.
DOCUMENTATION: UGH.Tty.

VIEWDATA: A subsystem to display on the Alto screen three-dimensional data stored as a two-dimensional array of single-word values.
DOCUMENTATION: ViewData.Tty.

VIEWIC: A part of the ICARUS2 System that displays the data created by Mike on the Alto screen, simulating the actions of the pattern generator.
DOCUMENTATION: [Maxc]<ICARUS>ViewlcDoc.Press.

VPRINT: A subsystem to output text files such as .TTY, UGH, BRAVO, or GYPSY, to a Versatec printer.
DOCUMENTATION: Under development.

FUNCTIONAL CROSS REFERENCE

The following list of Alto subsystems is organized according to the general function they perform. Because many subsystems perform more than one function or a function may be thought of in a variety of ways, an item may be listed more than once.

The major functional headings are:

DOCUMENT CREATION
FILES
FONT CREATION
RECOVERING

HARDWARE DESIGN
HARDWARE DIAGNOSTICS
HARDWARE DRIVERS

MESSAGES
PRINTING
PROGRAMMING

DOCUMENT CREATION

EDITORS

TEXT

BRAVO: Rich in formatting features.
GYPSY: Features to handle groups of files (e.g. chapters or modules).
PROOFREADER: Produces an output file of questionably spelled words.
QED: A line editor.
UGH: An in-core editor that uses the keyset for command input.

GRAPHIC

DRAW: Pictures composed of lines, curves, text and smoothed mouse tracks.
FRED: A Spline editor for font work.
MARKUP: Dot pictures of lines, areas, text, and mouse tracks.
SIL: For creating diagrams composed of lines with text captions.

IMAGES

AIS: Image manipulation, printing, Press file creation.

PAGE MAKEUP

MARKUP: Create new or move pre-existing Press files along side existing text.

MERGING

AIS: Merges bitmap files (e.g PRESS output and AIS files).
EMPRESS: Append press files to personalized coversheets.
PRESSEEDIT: Generate Press files from pages of other Press files.

PRINTING: see **PRINTING** below.

FILES

DISPLAY

AISSHOW: AIS files.
OEDIT: Alto files in octal.

MARKUP: Press files.
PRESS: Press files.
READPRESS: Press file internals.
SHOWAIS: 8 bit/pixel AIS files from remote file servers.
TYPE: Text files.
VIEWDATA: Three-dimensional information stored as a matrix of values.

TRANSFER

COPYDISK: Copies whole disks, Diablo Models 31/44 and Trident T-80/T-300.
FTP: Copies a file between Altos, Alto-File Server, and Alto-Maxc.
GYPsy: Transmits its files to a Communicating 800 ETS.
NEPTUNE: Copies files between disks of a dual drive Alto.
PUT: Copies files between disks of a dual drive Alto.
SIGMA: Copies files between Alto and SIGMA 3.
TFU: Copies files between Trident drives.

ALTO FILE SYSTEM

CALLFTP: Very small version of FTP.
CLEANDIR: Garbage collect disk directory.
CREATEFILE: Adds new file of specified size on consecutive pages, if possible.
CRUMPLE: Compresses and optionally encrypts a file.
DDS: Large directory manipulation subsystem.
EXECUTIVE: Delete files, list directory.
FIND: Locates and displays lines containing a specified text string.
FTP: Transfers files between Alto and Alto, a File Server, or Maxc.
NEPTUNE: Small directory manipulation subsystem. Deletes, renames, and copies files between disks of a dual drive Alto.
OEDIT: Display and modify file in octal.
PUT: Deletes, renames, and copies files between disks of a dual drive Alto.
SCAVENGER: Checks and corrects the disk.
SIGMA: Transfers files between an Alto and SIGMA 3.
SORT: Sorts up to 1000 items delimited by carriage returns.
TYPE: Displays contents of text files.

FILE SERVER SYSTEM

IFS: The server.
IFSsCAVENGER: Checks and corrects Trident T-80/T-300 disks.
TFU: Initialize directory and verify disk.

TRIDENT FILE SYSTEM

IFSSCAVENGER: Checks and corrects Trident T-80/T-300 disks.

TFU: Initialize and list directory, verify disk, copy and delete files.

RECOVERY

IFSSCAVENGER: Check and correct disks of File Server and Trident file systems.

SCAVENGER: Check and correct disk of Alto file system.

FONT CREATION

SPLINES

DRAW: Create splines using mouse or knots.

FRED: Create and edit splines, create font files.

BITMAPS

PREPRESS: Scales and rotates splines, converts to and edits bitmaps.

DEVICE FORMATS

PREPRESS: Creates printer and display fonts from bitmaps.

HARDWARE DESIGN

CIRCUIT BOARD

SIL: Create and edit logic diagrams.

ANALYZE: Converts SIL drawing for GOBBLE, generates SIL of unassigned pins.

GOBBLE: Generates wirelist and routing information.

BUILD: Aids data management aspects, keeping data files current.

INTEGRATED CIRCUIT

ICARUS2: Layout integrated circuits.

IFD2: Generates human readable form of ICARUS2 files.

MIKE: Transforms ICARUS2 file to form for Mann 3000 pattern generator.

TRANSFILE: Generates human readable form of MIKE file.

VIEWIC: Displays MIKE output.

ICGERB: Generates Gerber photoplotter output from ICARUS files.

HARDWARE DIAGNOSTICS

USER

CRTTEST: Displays a rectangular grid. (.boot/.run files).
DMT: Memory diagnostic that transmits results to PEEK. (Alto/server boot files).
KEYTEST: Keyboard diagnostic. (.boot/.run files).
MADTEST: Diagnostic for RAM, ALU, and emulator. (.boot/.run files).

INSTALLATION

DIEX: A Diablo disk exerciser similar in operation to TRIEX.
DISKTEST: Bootfile diagnostic for the Diablo Model 31.
EDP: Ethernet interface diagnostic.
ORBITTEST: An ORBIT Interface diagnostic.
PEEK: Collects PeekReport/EventReport packets on the file Peek.Reports.
PEEKSUM: Summarizes DMT error reports collected by PEEK on the file PeekSummary.Tx.
PUPTEST: Ascertains status of network servers.
TRIEX: A Trident T-80/T-300 diagnostic.

HARDWARE DRIVERS

See the Alto Hardware Catalog under appropriate device.

MESSAGES

SENDING

CHAT: Accesses SNDMSG on Maxc.
LAUREL: Display-based, MSG compatible, runs on the Alto.

RECEIVING

MAILCHECK: Interrogates Maxc for new mail.
CHAT: Accesses MSG on Maxc to retrieve mail.
LAUREL: Display-based, MSG compatible, runs on the Alto.

PRINTING

REMOTE

REFORMATING

PREPRESS: Generates a Press file from .Tty and many .Press files.

PRESS

BRAVO: Sends the text file to Spruce.
EMPRESS: Sends the specified Press and/or Text files to Spruce.
HARDCOPY: Prints any file any place.
GYPSY: Sends the workfile to Spruce.
SPRUCE: A Press server for Dover/Sequoia/Pimlico/Penguin or Puffin.

LOCAL

BRAVO: Prints text files on the attached Diablo HyType.
DPRINT: Prints text files on the attached Diablo HyType.
GYPSY: Prints the workfile on the attached Diablo Hytype.
PRESS: Prints Press files on slot and Versatec printers.
VPRINT: Prints text files on Versatec printers.

PROGRAMMING

EDITORS

See DOCUMENT-EDITORS-TEXT above.

ASM/BCPL

ASM: Alto machine language assembler.
BCPL: BCPL compiler.
BLDR: Loader for ASM and BCPL relocatable files.
BUILDBOOT: Generates a type B bootfile.
SWAT: an emulator level, BCPL oriented debugger.
INSTALLSWAT: Installs SWAT on a disk.
LISTSYMS: Converts .SYMS files to human readable form.

MU

MU: Microcode assembler.
RAMLOAD: Loads RAM with MU output.
PACKMU: Converts MU output for RPRAM.
RPRAM: Loads RAM with RPRAM output.

DEBUGGING AIDS

AISDUMP: Writes out pixel values in decimal.
CONDENSE: Recovers display bitmap from SWAT or SWATEE file.
SWAT: An emulator level, BCPL oriented debugger.
LISTSYMS: Converts .SYMS files to human readable form.
BTREESTEST: A B-tree dictionary maintenance program.
PEEKPUP: Peeks at PUPs going to and from a specific Ethernet address.
PUPTEST: Interacts with new subsystems that use the PUP protocol.
READPRESS: Displays entities within a Press file.

RECOVERING

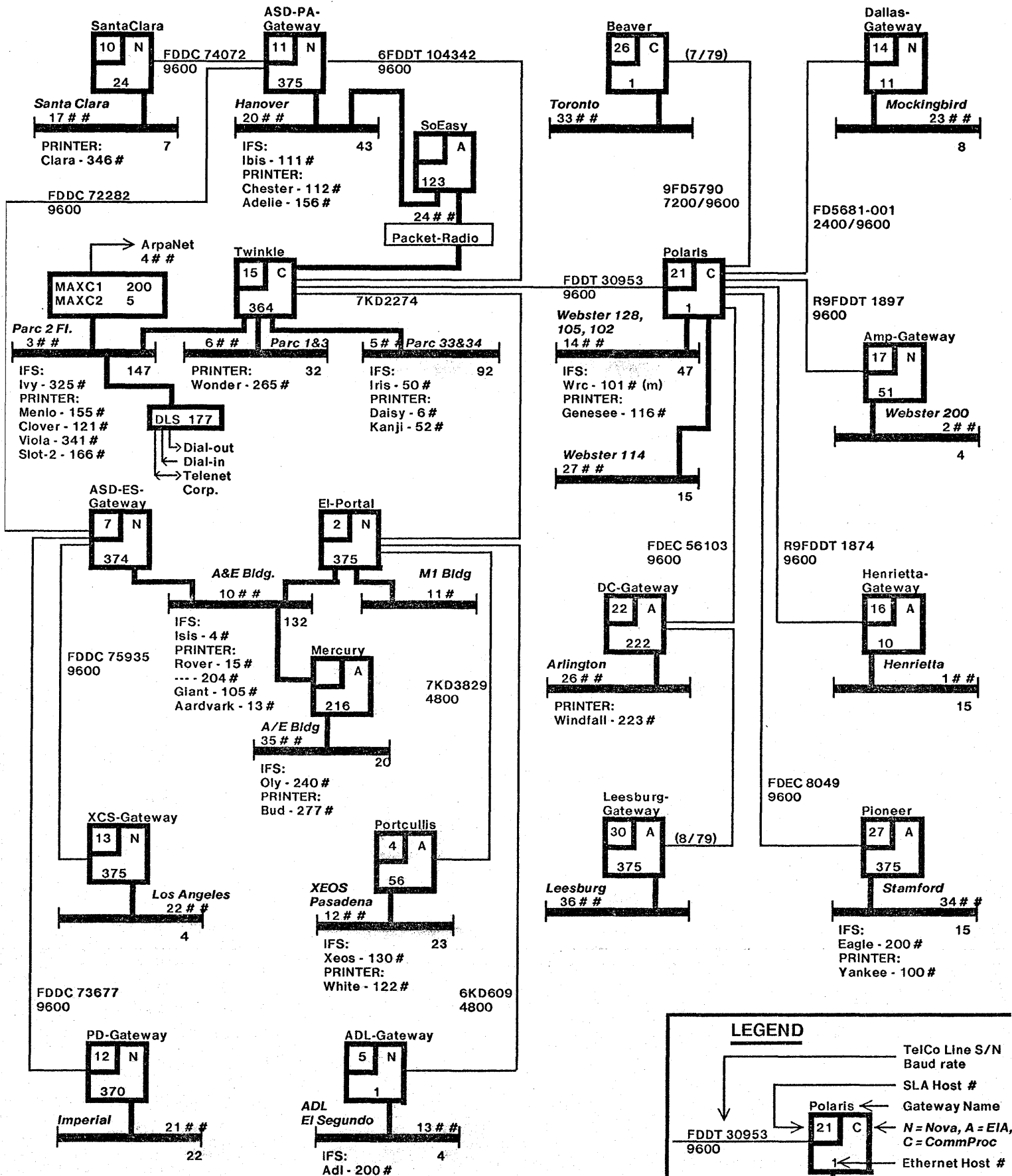
DISK FAILURES

See FILES-RECOVERY above.

SUBSYSTEMS FAILURES

BRAVOBUG: Recovers BRAVO files to point of failure.

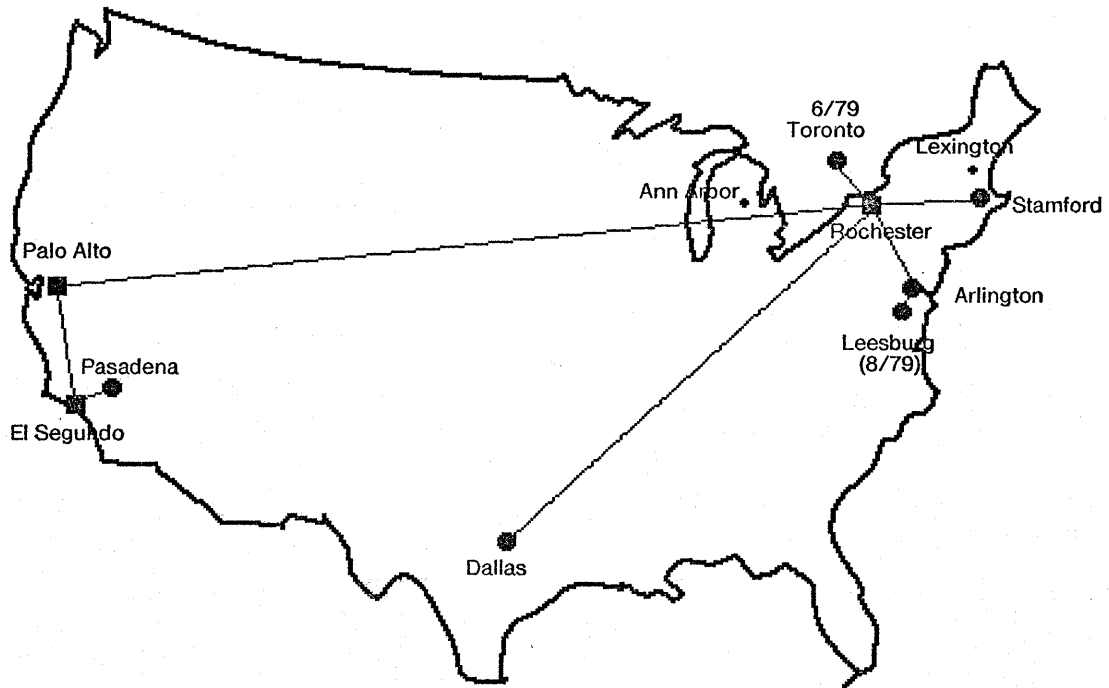
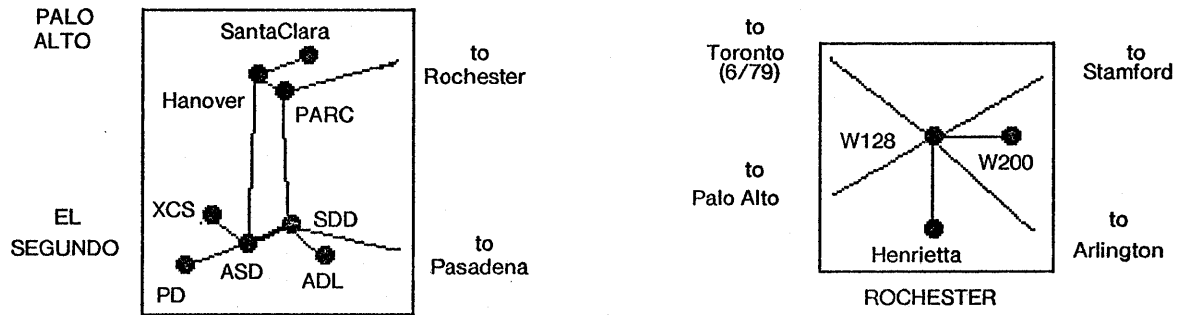
CONDENSE: Recovers display bitmap from SWAT or SWATEE file.



ALTO NATIONAL NETWORK NETWORK TOPOLOGY
July, 1979

LEGEND

- TelCo Line S/N
- Baud rate
- SLA Host #
- Gateway Name
- N = Nova, A = EIA, C = CommProc
- Ethernet Host #
- Location
- Network #
- Hosts on Net
- (m) => mail server



NATIONAL ALTO NETWORK
June, 1979

Filed on: [maxc]NetworkMap.press

COMPLETE ALTO DRAWINGS

Revised 7/18/79

<u>Drawing No.</u>	<u>Rev.</u>	<u>Description</u>
209643	A	Assembly, Terminator (Modification)
209911	G	Assembly, Display
209913	C	Assembly, Connector Panel (Display)
209915	A	Specification, Procurement - Monitor
209916	A	Specification, Procurement - Case, Monitor
209919	B	Specification, Procurement - Keypad
209920	B	Specification, Procurement - Mouse
209926	B	Specification, Procurement - Ethernet Transceiver
209939	B	Specification, Procurement - Disk Drive
209962	B	Assembly, Keypad
209973	G	Subassembly, Keyboard
209998	G	Specification, Procurement - Keyboard Button
216252	B	Assembly, PW - Alto Trident Disc Interface
216273	C	Assembly, PW - 32K x 10 Storage Module
216284	D	Assembly, Backwiring Board
216299	B	Assembly, Control Cabinet
216302	F	Assembly, Processor Chassis (DOCUMENT BEING REVISED)
216305	B	Assembly, Backwiring Board (Wired)
216312	E	Assembly, PW - Memory Data Interface
216323	G	Assembly, Printed Wiring - Ethernet
216339	E	Assembly, Printed Wiring - Display Control
216347	G	Assembly, PW - Memory Address Interface
216362	G	Assembly, Alto II System
216363	B	Assembly, Front Cabinet Cover
216364	A	Assembly, Rear Cabinet Cover
216365	F	Assembly, Printed Wiring - CRAM
216373	D	Assembly, Printed Wiring - Control Board
216381	F	Assembly, PW - Arithmetic Logic Unit
216389	J	Assembly, Printed Wiring - Disk Control
216400	B	Assembly, Cable - Disk (Internal)
216401 (Charted)	A	Cable, Modification - Ribbon (Disk Internal)
216402	B	Specification, Procurement - Cable
216405	C	Assembly, Cable - Keyboard (Microswitch) (External)
216407	F	Assembly, Cable - Keyboard (ADL) (Ext. Cable) (Second Generation)
216408	D	Assembly, Cable - Keyboard (Internal) (ADL)
216410	C	Assembly, Cable - Display (External)
216411	D	Assembly, Cable - Ethernet (External)
216412	D	Assembly, Cable - Display (Internal)
216413	D	Assembly, Cable - Ethernet (Internal)
216414	D	Assembly, Cable - Keyboard (Internal)
216415	C	Assembly, Cable - CRAM/RAM (Internal)
216416	G	Assembly, Cable - Hy Type/XREG (Internal)
216420	G	Assembly, Printed Wiring - Keyboard
216421	B	Assembly, PW - Terminator Module
216425	B	Assembly, Power Supply Harness
216432	A	Assembly, Cable - Fan
216433	A	Specification, Figure Dial
216438	C	List, Wire - Backwiring Board
216440	C	Assembly, Cable - Hy Type (External)
216445	C	Assembly, Spring & Sensing Pin Stack (Keyboard)
216448	A	Assembly, Hytype II Printer

216450	B	Assembly, Cable - Versatec Printer/Plotter Interface
216451	A	Specification, Procurement - Hytype II Printer
216452	A	Specification, Procurement - Printer/Plotter
216466	B	Assembly, Cable - Tricon Radial (Internal)
216467	A	Assembly, Cable - Tricon Bussed (Internal)
216468	I	Assembly, Printed Wiring - Trident Disc Interface
216480 (Charted)	E	Assembly, Cable - Tricon Radial (External)
216483 (Charted)	C	Assembly, Cable - Tricon Bussed (External)
216484	B	Assembly, Printed Wiring - Control Board (2K)
216492	F	Assembly, Printed Wiring - Orbit Input Module
216500	D	Assembly, Printed Wiring - Orbit Control
216508	D	Assembly, Printed Wiring - Orbit Output Module
216518	D	Assembly, Printed Wiring - Orbit Memory
216536	C	Assembly, Printed Wiring - Relay Module
216537	B	Assembly, Control Panel
216542	D	Assembly, Cable - Tricon Radial (External)
216545	A	Assembly, PW - Extender Module
216549	C	Assembly, Printed Wiring - Motor Driver
216554	F	Assembly, Printed Wiring - Video Adapter
216559	D	Assembly, Printed Wiring - Command Adapter
216564	H	Assembly, Printed Wiring - Engine Control
216571	B	Assembly, Chassis - Card Cage
216572	D	Assembly, Printed Wiring - Mother Board
216577	B	Assembly, Printed Wiring - Extender Board
216578	A	Assembly, Cable-Orbit/Adapter
216579	B	Assembly, Cable - Transformer
216580	B	Assembly, Cable - Logic Power Supply
216583	A	Assembly, Printed Wiring - Display Intensity Control
216589	B	Assembly, Cable - 80 Volts Interface
216591	B	Assembly, Backwiring Board Alto II/Orbit (Wired)
216592	A	List, Wire-Backwiring Board (Orbit)
216593	C	Assembly, Orbit
216595	B	Assembly, Alto II/Orbit
216597	D	Assembly, Cable - Modulator Driver
216598	C	Assembly, Cable - Laser Power Supply
216599	C	Assembly, Top Harness Printer
216600	C	Assembly, Printer (Dover)
216601	B	Installation Dwg, Dover System
216602	C	Assembly, Bottom Harness Printer
216603	A	Assembly, Printer (Modification)
216604	B	Assembly, Plate - Control Panel Mounting
216605	B	Assembly, Cover - Printer (Right)
216606	B	Assembly, Cover - Printer (Left)
216607	C	Specification, Procurement, Slot Head Assembly
216614	A	Specification, Procurement - Data Terminal Desk
216617	A	Assembly, Front Console Cover
216620	B	Assembly Alto Orbit System (Console)
216621	H	Assembly, Printed Wiring - Trident Multiplexer
216630	B	Assembly, Cable - Paper Tray Noise Suppressor
216633	C	Assembly, Alto II/Orbit Conversion
216635	B	Assembly, Cable - Tricon Radial to Mux (Internal)
216636 (Charted)	D	Assembly, Cable - Trident Mux (Internal)
216637	A	Assembly, Tricon Kit
216638	C	Assembly, Tricon Mux Kit
216646	B	Assembly, PW - Memory Extension & Terminator Mod.
216652	C	Assembly, Backwiring Board (Wired) (Extn. Mem.)
216695	B	Assembly, PW - Control Board (1K) (Extn. Mem.)

217113 A List, Wire - Backwiring Board (Extn. Mem.)
 217160 A Assembly, Fan Mtg Tray
 217166 B Assembly, PW-Keyboard (Alto II/Microswitch)
 217172 A Assembly, Keyboard - Alto II/Microswitch (Low Profile)
 217173 E Assembly, PW - Memory Address Interface (XM)
 217174 C Assembly, PW - Memory Data Interface (Extn. Mem.)
 217175 B Assem.....
 D Assembly, Printed Wiring - CRAM (Extn. Mem.)
 217178 A Assembly, Control Cabinet (Extn. Mem.)
 217179 A Assembly, Alto II System (Extn. Mem.)
 217180 A Assembly, Processor Chassis (Extn. Mem.)
 217181 A Assembly, Cable - Keyboard (Alto II/Microswitch)
 217182 A Assembly, Cable - Photocell
 217183 A Assembly, TTL Ros Adapter (Dover II)
 217184 B List, Wire - Backwiring Board (Orbit Extn. Mem.)
 217185 A Assembly, Backwiring Board Orbit (XM) (Wired)
 217187 A Assembly, Printed Wiring - 64K Storage
 217188 A Assembly, Printed Wiring - 128K Storage
 217189 A Assembly, Printed Wiring - 256K Storage
 217288 A Assembly, Printed Wiring - Extender Board
 217635 A Assembly, Alto II/Orbit (XM)
 217636 A Assembly, Orbit Extended Memory Conversion

Filed on: <CWILLIAMS>AltoDwgs.Press

XEROX

Whole ALTO World Newsletter

For Xerox internal use only

CONTENTS

SPECIAL ANNOUNCEMENTS	1
Change in WAW Meeting	1
Alto's Give Way To DO's	1
What Will We Call the WAW???	1
GENERAL NOTES	2
Nova Gateway Recall	2
Electronic Mail System Changes	2
Alto Ethics	4
Another Laurel Hack	5
Ordering Disk Packs in El Segundo	5
New Network Diagram	5
Alto Network Topology Diagram	37
University Grant Notice	5
MARKET PLACE	6
Hardware Catalog Update	6
1600 BPI Mag Tape Interface	6
New Software Offering	7
TOOLS	7
Hardware	7
Maintenance Notes	8
Software	8
New Releases	8
Re-Releases - Subsystems	8
Brownie	11
Re-Releases - Packages	9
TECHNOLOGY	10
Future Trends in Electronic Publishing	20
Micrographic and Personal Computers in Information Science in 2001	29
CATALOGS	10
Alto Hardware Catalog	15

XEROX

Whole ALTO World Newsletter

For Xerox internal use only
Published and edited by Ron Cude
Message <CUDE> or call Intelnet 8*823-2465

TECHNOLOGY AND TOOLS FOR THE FUTURE

August 30, 1979

SPECIAL ANNOUNCEMENTS

CHANGE IN WAW MEETING

The date for our next meeting has been changed from October 2 to Tuesday October 9, 1979. The meeting will be held at the TLC Best Western Hotel, 460 Totten Pond Road, Waltham, Mass., 617-890-7800. Darwin Newton, our host, has provided a list of other hotels in the area:

Holiday Inn Waltham, Totten Pond Rd., Waltham, 617-890-3000 (adjacent to the Best Western)
Waltham Motor Inn, 385 Winter St., Waltham, 617-890-2800
Sheraton Lexington, 727 Marrett Rd., Lexington, 617-862-8700
Marriott Hotel, 2345 Commonwealth Ave, Newton, 617-969-1000
Holiday Inn, 399 Grove St., Newton (Riverside), 617-969-5300
Wellesley Motor Inn, Rt. 9, Wellesley, 617-235-8555
Howard Johnson, 300 Washington St., Newton, 617-969-3010 (over turnpike)
Boston Sheraton Hotel, 39 Dolton St., Boston, 617-236-2000 (downtown)
Copley Plaza Hotel, Copley Square, Boston, 617-267-5300
Colonnade Hotel, 120 Huntington Ave., Boston, 617-261-2800

WRC will have a series of demonstrations available Wednesday afternoon, October 10, for those who would like to see what's new at Webster. Make your plans now to attend. Please message Ron Cude to reserve a place at the meeting.

ALTO'S GIVE WAY TO DO'S

Well Alto fans, it looks as though the current Alto build will really be the last one ever. I know you've heard this before, but this time it looks for real. The flavor of DO that is likely to be the standard configuration will be the CSL/ASD package. More detailed information about this will be known soon and passed on in the Newsletter.

WHAT WILL WE CALL THE WAW???

Now that DO's will be augmenting, and at some point in time outnumbering the Alto's that make up our Whole Alto World, the time has come to seriously think about changing our name to reflect more than just the Wonderful World of the Alto. Perhaps a contest. Submit your ideas, good bad or ugly, to Ron Cude. The winner may receive a no expenses paid trip to beautiful downtown Gilroy, Ca., "The Garlic Capital of the World" and \$1.47 in walking around money.

WHOLE ALTO WORLD NEWSLETTER

GENERAL NOTES

NOVA GATEWAY RECALL

PARC, SSL has decided to reclaim the Nova Gateways that it has on loan to various groups. This decision has been motivated by two major factors:

1. Many of the pieces of hardware in these gateways are needed by SSL.
2. The existence of Alto Gateways will soon obsolete the Nova Gateway. The Alto Gateway now dominates the Nova Gateway in terms of its reliability and capabilities. Soon the Nova Gateway hardware will not be supported by the gateway software.

Here are the details:

PARC cannot guarantee software compatibility with any Nova Gateway past January, 1980. SSL requires the return of all Gateways on or before July 1, 1980.

Since we are all at 1980 Operating Plan time, it is important that you factor the replacement of your Nova Gateway into your plans.

Submitted by Bert Sutherland, PARC SSL

ELECTRONIC MAIL SYSTEM CHANGES

This item concerns some important changes in the operation of our electronic mail system. The changes will be most noticeable to Laurel users, though users of Maxc message systems (Msg, etc.) will also be affected somewhat.

Until now, all users have had mailboxes on one central mail server machine, namely Maxc. All Laurel mail delivery and retrieval has consisted of making a connection to Maxc. As the number of Alto users outside Palo Alto has grown, this arrangement has become administratively unmanageable, as well as being a source of poor performance.

We are therefore in the process of converting over to a new arrangement, under which nearly all users outside Palo Alto will keep their mailboxes on a local mail server machine (an IFS) rather than on Maxc. When you deliver or retrieve mail, you will make a connection to your local mail server rather than to Maxc. The mail servers will take care of delivering messages to other mail servers when necessary. This arrangement is expected to provide much better service to non-Palo Alto users, and the administration of the mailbox accounts will be decentralized.

Each user community has a name, called its "registry", that identifies the mail server that services it. Eventually, the registries will be as follows:

PA (Palo Alto)
 ES (El Segundo)
 WBS..... (Webster area)
 HENR (Henrietta area)
 EOS (Electro-Optical Systems, Pasadena)

That is, all users in the Palo Alto area will keep their mailboxes in a registry called PA (which is really just an alias for Maxc); all users in the El Segundo area will keep their mailboxes in a registry called ES (a new IFS to be installed for this purpose); etc.

For now, we are just going to use the PA, WBST, HENR, and EOS registries. Users who currently have their mailbox on the Erie IFS will be in either WBST or HENR, users who currently who have their mailboxes on the XEOS IFS will be in the EOS registry, and users who currently have their mailboxes on Maxc1 or Maxc2 will be in the PA registry.

WHOLE ALTO WORLD NEWSLETTER

Later, the ES registry will be created, and certain PA users' mailboxes moved to ES, **WBST**, **HENR**, and **EOS**, as appropriate. But for the next few weeks, all users in El Segundo will continue to have their mailboxes in the PA registry. A general announcement will be made at the time the El Segundo mailboxes are moved to the ES registry. **No MAXC mailboxes will be removed until another announcement is made.**

Because there are multiple registries, when you send a message to somebody outside your own registry, you must specify the registry of that recipient. This is done by appending a period followed by the registry name. For example, if your name is Jones and your registry is PA, and you want to send a message to Smith whose registry is **WBST**, you will have to say:

To: Smith.WBST

When Laurel sends the message, it will identify the message as coming

"From: Jones.PA".

When you send a message to somebody in your own registry, you do not need to qualify the recipient's name with the registry name. That is, if you are Jones and your registry is PA, and you want to send a message to Brown whose registry is also PA, you need only say:

To: Brown

(though it is also correct, though unnecessary, to say "Brown.PA"). Laurel's "Answer" command will always qualify the automatically-generated recipient names with registries wherever appropriate.

If your mailbox is currently on MAXC, and you are a Laurel user, here is what you should do right now. Use the "Get" menu item to read the file Laurel.profile into the composition window. In this file, you will find the following statements:

**Send: Maxc
Retrieve: Maxc
Authenticate: Maxc**

Delete these three statements and replace them with the single statement:

Registry: PA

Now use the "Put" menu item to write onto Laurel.profile, then "Quit" and restart Laurel.

This enables all the "multiple registry" mechanisms in Laurel. Until you have done this, Laurel will refuse to send messages to recipients in other registries.

You **MUST** be running Laurel 5 in order for this to work. If you have not already converted to Laurel 5, you should do so now, by retrieving <Laurel>Laurel.cm from your local file server and issuing the executive command "@Laurel@".

If you are outside of Palo Alto, you will be notified by your local support organization of the time at which your mailbox will be moved from Maxc to your local IFS mail server. This will be a few weeks from now (the exact details of this operation are still being worked out). At that time, you will be asked to edit your Laurel.profile again, to change your Registry from PA to your local registry name.

If your mailbox is currently on MAXC2, and you are a Laurel user, you should proceed as above, except that the three original statements in Laurel.profile should be replaced by:

**Registry: PA
Retrieve: Maxc2**

WHOLE ALTO WORLD NEWSLETTER

If your mailbox is currently located on your local IFS (Erie or XEOS) rather than on Maxc, you should follow the procedure given above, except that the "Registry" statement should say:

- Registry: **WBST** (if you are in Webster)
- or
- Registry: **HENR** (if you are in Henrietta)
- or
- Registry: **EOS** (if you are in Pasadena)

You will now be able to send messages to people in other registries, whereas before you could send only to people who had mailboxes on your local IFS.

You might not know that there are already sizable mail user communities in **WBST** and **EOS**. Until now, they have been independent from the Maxc-based mail system, and it hasn't been possible to send messages between these communities. You can now send messages directly to people in these registries, rather than indirectly via **PA** mailboxes such as <WRC> and <LizBond>.

If you maintain any distribution lists, particularly any that are used in multiple organizations or whose membership crosses organizational boundaries, you should modify them to include the correct registry of each recipient. Centrally-maintained distribution lists for Parc and SDD will have this done to them automatically.

To send a message to the Arpanet, use the syntax "**name@host**" or "**name at host**", where "host" is the Arpanet host name. The "@" or "at" construct is used exclusively for Arpanet host names, and Laurel knows that any message with a recipient name containing "@" or "at" must be sent to Maxc for forwarding to the Arpanet.

If you receive messages from the Arpanet and your mailbox is not in the **PA** registry, you will have to inform your Arpanet correspondents that they must address messages to you by specifying "**name.registry@Parc-Maxc**". For example, if your name is Smith and your registry is **ES**, then messages from the Arpanet will reach you only if they are addressed to "**Smith.ES@Parc-Maxc**". (When you send a message to the Arpanet, Laurel will correctly identify you in this form.)

If you use Msg (or some other Maxc mail system) rather than Laurel, you should be aware that the recipient names put in the "To" and "cc" lists by the "Answer" command won't always be properly qualified by the registry name. These mail systems aren't maintained here, and there is nothing we can do about this problem.

The Maxc/IFS distributed mail system is still very much an "interim" arrangement. In future mail systems, registries will be decoupled from file servers entirely. Accomodating such future changes is the reason for choosing registry names ("**PA**", "**EOS**", etc.) that are distinct from file server names. Please use registry names, not file server names such as "Maxc", when addressing messages.

If you have any questions or problems, please contact your support organization.

Submitted by Ed Taft, PARC CSL

ALTO ETHICS

There has been little if any discussion about the philosophy regarding the privateness of files on the Alto Network, particularly those on easily accessible shared utilities such as MAXC and IFS's. Many personal directories allow read access to the World. Is it ethical to assume that this implies permission for anyone to browse? The temptation to do so can be almost irresistible. Yet we are not so tempted to browse through the paper documents in someone else's unlocked file cabinet.

WHOLE ALTO WORLD NEWSLETTER

Probably several other ethical issues could also be found. For example, is it OK to use an Alto for personal (not-for-profit) activities outside of business hours? How about IFS file space (since there is presently a surplus), or Dover? How about MAXC or other local computers, during periods when they are otherwise idle?

This should be actively debated within the Whole Alto World. When the Alto community was small and cohesive, common behavior patterns diffused by osmosis. But now that the Whole Alto World includes at least a thousand people, more explicit indoctrination may be necessary.

Submitted by Dave Damouth, WRC

ANOTHER LAUREL HACK

It has been observed that many Laurel users make good use of the "Move To" mail file feature to organize their mail into different categories. This is as it should be except that when the list of different mail files gets very large, it becomes difficult to remember them all or what they are called. Thus in order to recall a mail file name, it becomes necessary to go back to the Executive and get a list of all the "*.mail" files and then go back into Laurel to retrieve the desired one.

One way to overcome this is to create a message that lists all of your mail files so you can refer to it while in Laurel. To do this, display "New Form", compose the message listing all the names of your mail files, select the "Put" command and edit in the brackets an easy name like "MailFiles" and type ESC. This writes your message on your Alto disk. To retrieve this message while in Laurel, select "Get", type "MailFiles" in the brackets and ESC. This message can be edited at any time to reflect maintenance of your list of mail files and then re-"Put". It can also be displayed, edited, and re-"Put" using Bravo. Note that it is not necessary to ever "Deliver" your mail file list message.

Submitted by Chuck Henderson, ASD

ORDERING DISK PACKS IN EL SEGUNDO

Alto disk packs are now available from *Forms and Supplies* in El Segundo. Please direct requests for ALTO disks to *Forms and Supplies*, Mail Stop M4-14, INTELNET 8*823-2378.

You may order ALTO disks (Memorex model MK III 2F-12) by sending a Supplies Request form (#339) and specifying stock number MK III 2F.

ASD will no longer stock ALTO disks. If you have any questions, contact:

J. H. Mathus
WOS/Graphic Services
El Segundo, M4-04
INTELNET 8*823-2757

NEW NETWORK DIAGRAM

A new Alto Network Topology diagram is available and is stored in [MAXC]KAltoDocs> AltoNetwork.press and is included within the Newsletter, page 37.

UNIVERSITY GRANT NOTICE

As some of you may know, Xerox is making a grant of Alto hardware and software to the computer science departments of three universities: *Carnegie-Mellon*, *MIT*, and *Stanford*. The *University of Rochester*, which already has some Alto equipment, will also participate in the software release. Included in this grant will be nearly all the "basic" software used at PARC for document preparation and for Bcpl and Mesa programming.

WHOLE ALTO WORLD NEWSLETTER

The mechanism for release will be that representatives of the universities copy files from certain designated directories on Maxc1, including <Alto>, <AltoDocs>, <AltoSource>, <Mesa>, and <Pup>, as well as several new directories that are in the process of being set up. Maxc has the protection facilities required to permit us to give the university liaisons access rights to selected directories and to selected files within those directories; the security of other files on Maxc will not be compromised by this release procedure. The university liaisons will have no access at all to file servers besides Maxc.

The policy being followed with respect to this software release is that we are making available as much software as possible, consistent with proprietary considerations. Exceptions will be decided on a case-by-case basis by a Xerox committee that is administering the grant program.

The software release is being coordinated by PARC liaisons who are alumni of the universities (presently Roy Levin (CMU), Mike Schroeder (MIT), Dan Swinehart (Stanford), and Joe Maleson (U of R)). In general, all software and documentation in the above-mentioned directories (either now or in the future) will be released to the universities; however, all files are being considered on a case-by-case basis, and authors or maintainers will be consulted before any software is released. Once a given piece of software or documentation has been cleared for release, updates to or new versions of it will also be made available to the universities unless the PARC liaisons are notified to the contrary.

For legal reasons, each source file and document released as part of the grant program must be edited to include the following copyright notice, near the beginning of the file:

Copyright Xerox Corporation 1979

This will be taken care of by the PARC staff members who are coordinating the release. However, next time you modify such a program or document for which you are responsible, you should be careful to obtain fresh copies of the source files from Maxc rather than use versions you might have saved elsewhere (e.g., on an Alto disk).

If you are responsible for software or documentation kept in the above-mentioned Maxc directories, and you have not yet been contacted about this matter (or you have questions or comments), you should contact Dan Swinehart, who is coordinating the software release at the present time.

Submitted by Ed Taft, PARC CSL

MARKET PLACE

Market Place provides a forum for Alto users to make offerings and requests for Alto related hardware and software. To place an "ad" or offering, send the text to the WAW Coordinator, Ron Cude, in El Segundo, message <CUDE>, or phone Intelnet 8*823-2465.

HARDWARE CATALOG UPDATE

The Alto Hardware Catalog is still being updated. (Primarily because I've received very little input from any of you regarding new items.) Please provide me with any items eligible for inclusion. Devices included in this catalog have at least one prototype considered to be working properly and must be reproducible from existing documentation. A copy of the current (outdated) catalog (see page 15) is in the Newsletter. Please send comments, additions and corrections to Ron Cude.

1600 BPI MAG TAPE INTERFACE

Alto II's can now run 1600 BPI 9 Track Phase Encoded (PE) Magnetic Tapes. An installation requires two ASD tape controller boards, cables, and a few backplane modifications. Both

WHOLE ALTO WORLD NEWSLETTER

Kennedy (ANSI) and Mohawk style drives can be accomodated, with up to four drives daisy-chained by one controller set. The initial software has been released. This includes Tape (integrated with Trident) microcode, low level (buffer I/O) software, a streams package, a utility for copying unlabeled tapes of varying formats, and a diagnostic. Interested parties should contact Alan Paeth % <LIZBOND.EOS>, XEOS MS - 359 or Intelnet 8*844-2232.

Submitted by Alan Paeth, XEOS

NEW SOFTWARE OFFERING**★ AFTP.RUN ★**

Requirements at ADL have necessitated the production of a modified version of EFTP, called AFTP.Run. It is compatible with the old EFTP and uses modified versions of the latest EFTP packages.

It differs from EFTP in that it caters to a new EFTP package type: a LeaderPage packet. If requested by a /f global switch, AFTP will send the leader page of the file being transmitted as a separate LeaderPage packet. The receiving AFTP will recognize this special packet and try to open a new file under the name carried by the LeaderPage packet. Subsequent data packets will be written to this file. If the file already exists and the receiving version of AFTP was not started with the /f global switch, then the file will not be accepted and an abort code will be returned to the sender. This prevents accidental overwriting of files.

The format of the command line differs slightly from EFTP in that the filename(s) to be sent are the last things on the line; i.e. the format is Aftp[/f] to/from foreign-port filename-list

The reason for the changes arise from a requirement to run a press printer server for the TC200 instead of a Spruce Server. Some way was needed to send file names to that server and still be able to do hardcopy (no file names) from Bravo, Laurel etc. Hence the compatibility requirement. The program is filed as [MAXC]<Cronshaw>Aftp.Run

Two other programs used regularly at ADL which might be useful to others are:

★ FILEDUMP.RUN ★

This program produces on the display and, optionally as dump.txt file, in Octal or Hexadecimal and Ascii form, a dump, disk page by page, of any file. It is used as a debugging aid when trying to generate or interpret files. The command line format is:

FileDump[/f] filename

If the /f option is used a file dump.txt is produced as well as the display output. This program is filed as [MAXC]<Cronshaw>FileDump.Run

★ SPLITFILE.RUN ★

This program is used to split large text files which are too big for Bravo. It arbitrarily splits the file into 60 page files called Part # .txt. It is filed as [MAXC]<Cronshaw>SplitFile.Run

TOOLS**HARDWARE**

SPG is currently debugging the first articles of the new 3K CRAM module and the Shugart Rigid Disk Controller module. They hope to be finished soon and go into the first build of production.

WHOLE ALTO WORLD NEWSLETTER**MAINTENANCE NOTES**

According to an article in *Computer World*, those occasional single errors that you see after DMT has run all night might be a problem that is inherent in the manufacturing of most large scale integration (LSI) memory chips.

It appears that the actual physical individual cell dimensions inside the chip are approximately 20 microns and store information as the "presence" or "absence" of 1,000,000 electrons.

All memory chips being manufactured today are made of ceramics and plastics which contain very small amounts of radioactive uranium and thorium. These emit alpha particles which occasionally pass through a bit cell. This can cause a disturbance of up to 1,500,000 electrons thereby destroying the information stored in the cell. It will remain this way until that bit has been rewritten.

This is why you may see from time to time a single error detected by DMT that does not repeat itself and appears to go away.

Submitted by Phil Hoffmann, Field Service

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local File Server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [ISIS].

NEW RELEASES**★ AIS ★**

Two new pieces of AIS software are being released at this time. They are AISHALFTONE.RUN and AISROTATE.RUN. Both are available from [Erie]<Ais> and both have been loaded with the new AIS software, i.e. to work with OS 17.

[Erie]<Ais>AisHalftone.Run, version 1.1, requires the text editable file Picture.Data. One is available on [Erie]<Ais>. The documentation can be found on [Erie]<Ais>AisHalftone.press. This software was written by Jim Bollman.

[Erie]<Ais>AisRotate.run, version 1.7, rotates bit/pixel or byte/pixel images. No documentation is available at this time. Written by Brett Fleisch.

RE-RELEASES - SUBSYSTEMS**★ SIL ★**

The Control-Shift-View command was broken and has been fixed. About 100 words of free space has been taken back and given to the operating system. This is required for the new OS 17 which will come out in the next month or so.

Corrections have been made on MCT6, added LS109, LS393, and 74128. In addition, the comments of DataSheets have been changed from Font-2 to Font-1-I. Files altered are Sil.lb5, Sil.lb6, TTLDict.Analyze, ECLDict.Analyze and TTL/ECL-DataSheets.

Additionally, the N123, retriggerable multivibrators, have been fixed so that ANALYZER will not complain about missing component name "a". Both [MAXC1] and [IVY] have been updated. File Sil.lb5 is in SilLibraries.dm.

WHOLE ALTO WORLD NEWSLETTER**★ BROWNIE ★**

Brownie 4.0, a program for moving files between several hosts (in particular updating directories on IFS's) has now been installed on the <Brownie> directory on [Iris]. To run Brownie, retrieve <Brownie>Brownie.image and <Mesa>RunMesa.run. Brownie requires about 2000 free pages on your local Alto disk to transfer large directories. There are a number of changes in this release. It is strongly recommended that you read the documentation (<Brownie>Brownie.press, see page 11 of the Newsletter) before using the new version. (The old version of Brownie has been moved to <OldBrownie>.) As always, bug reports and change requests should be submitted via the message system to <SDSupport>.

★ IFS 1.22 ★

Version 1.22 of IFS is now available on [MACX1]KIFS>. This is a maintenance release and changes have been limited to bug fixes.

A new startup-time global switch "/F" has been added: each occurrence of /F on the command line extends the directory file map by 100 words. This is needed only on [IRIS] and one /F elsewhere should be sufficient at present.

RE-RELEASES - PACKAGES**★ AIS ★**

The imminent arrival of OS 17 requires that all current AIS software be updated. Any software that is not updated will not work with OS 17. In addition, a small AIS file format change has been made at this time, and old AIS software will be unable to read the new format files. Updated software, however, will work on both OS 16 and OS 17 and will be able to read either old or new format AIS files.

To update your software, you need to retrieve the following files:

```
[ERIE]K[AIS]AisFile.d
[ERIE]K[AIS]AISUCA0.BR
[ERIE]K[AIS]AISUCA1.BR
[ERIE]K[AIS]AISUCA2.BR
[MAXC]K[Alto]TFS.DM
```

Recompile your sources, with the new AisFile.d, and load them with the new AISUCA*.BR files and the new TFS*.BR files.

Other AIS software being re-released is listed below:

```
[Erie]K[Ais]AIS.RUN -- version 3.2
[Erie]K[AltoDocs]AISMANUAL.PRESS (updated documentation, August 1979)
[Erie]K[Ais]AisAttributes.run -- version 1.2
[Erie]K[Ais]AisDump.run -- version 1.3
[Erie]K[Ais]AisHalfTone.run -- version 1.1
[Erie]K[Ais]AisHistogram.run -- version 3.1
[Erie]K[Ais]AisMagnify.run -- version 2.3
[Erie]K[Ais]AisMonochrome.run -- version 1.1
[Erie]K[Ais]AisRotate.run -- version 1.7
[Erie]K[Ais]AisShow.run -- version 2.3
```

WHOLE ALTO WORLD NEWSLETTER

TECHNOLOGY

This month two papers by Don Winter and Don Stewart are featured. *Future Trends in Electronic Publishing* begins on page 20. *Micrographics and Personal Computers in Information Science in 2001* begins on page 29.

CATALOGS

The Alto Hardware Catalog can be found this month beginning on page 15. Since it is being revised, please check it for errors and submit any changes or additions to Ron Cude.

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC] and [OLY]<AltoDocs>WawNewsM-YY.press or may be obtained from the editor, Ron Cude, ASD, by messaging <CUDE> or calling Intelnet 8*823-2465.

To	Brownie Users	Date	August 14, 1979
From	Bruce Malasky	Location	Palo Alto
Subject	Brownie 4.0	Organization	SDD/SO/Mesa

XEROX

Filed on: [IRIS]<BROWNIE>BROWNIE.BRAVO

Brownie is intended to aid in the problem of distributing software and maintaining consistent copies of *Bible directories* on several hosts. It may also be helpful in moving files around in private development directories during the software development process. This memo describes how to use Brownie and does not go into much detail about the actual implementation.

Getting Started

Brownie can be found on [IRIS]<BROWNIE> as BROWNIE.IMAGE and takes 242 pages on a Model 31. When Brownie is started a herald appears indicating which version of Brownie you are using and when it was built, followed by the current date and time. NOTE: Brownie 4.0 is written in Mesa 5.0 and is the only supported version available. Brownie 3.0 is a prerelease of 4.0 and is no longer supported. Version 4.0 represents major internal restructuring and cleanup from 3.0. Brownie 1.0 runs in Mesa 3.0 and only on a network with fewer than 20 nets. Brownie 2.0 was written in Mesa 4.2 and was never debugged. Any existing copies of this version should be deleted. The following command line is how Brownie is invoked.

```
>Brownie[/q] [DataFile]
```

If `DataFile` is not specified, Brownie searches the local disk for the file BROWNIE.DATA during initialization. The data file describes the operations to be performed. The `/q` switch will cause Brownie to prompt for connect passwords to directories as it parses the data file. It will always prompt for login passwords if they are missing. A sample data file and an explanation of its format are included at the end of this memo. When it is running, Brownie uses approximately 1 page on the local disk for every 15 files to buffer internal tables.

DataFile Options

Brownie can be used either to update one directory from another or to copy entire directories (all versions). While it is running, it displays (and logs in `mesa.typescript`) informational messages about what it is doing, including a list of the files it is transferring. The amount of information logged is controlled by a switch on the first line of the `DataFile`. The choices are: **error**, **brief**, **verbose**, **debug**; these cause Brownie to log successively more details. In **debug** mode, Brownie will print out passwords as well as other internal information. **Verbose** mode is worth trying once, to see all the steps that Brownie goes through. **Brief** mode is normally recommended, to keep the `mesa.typescript` from getting large. **Error** mode is not recommended, as Brownie will only display error messages. FTP error messages are followed by a code in parentheses; this is intended for debugging purposes only.

In Brownie 4.0, it is possible to specify a start-up time. This permits lengthy transfers that tie up a lot of network resources to be delayed until nighttime. Brownie processes the data file before doing any transfers so that any errors may be discovered immediately. While waiting for the start time, the display is turned off, and a moving Brownie cursor is displayed. The height of the cursor on the screen is proportional to the amount of time left to wait. In addition, it is also possible to specify a stopping time.

Files are moved using one of three methods: renaming, a diskless transfer, and retrieving followed by storing the file. The method to use is specified at the end of each command line with one of these keywords: **Rename**, **NoBuffering**, or **BufferFiles**; the default is **NoBuffering**. Renaming a file will work only if files are being moved within a directory, or if the user's login, as specified in the **DataFile**, grants the correct access permissions for both directories. In any event, renaming files across directories is only supported for two directories on the same IFS. **NoBuffering** indicates that the files are to be transferred using memory instead of the disk for a temporary copy of the file. This works by having one Ftp connection storing a file at the same time another one is retrieving it. **WARNING:** If the file transfer mode is **BufferFiles**, there must be enough room on the local disk to hold the largest file to be transferred. **NoBuffering** is about 30% faster than **BufferFiles**.

The parameter **TestFTP** now defaults to **TRUE** in order to help track down a suspected FTP problem. This will cause Brownie to trap to the debugger on a fatal FTP error in addition to creating the file **Brownie.FtpTrace** on the local disk. **TestFTP** must be explicitly set to **FALSE** to avoid trapping.

Brownie goes through some effort to do the right things for different hosts. For example, using "*" as a wild card on an IFS will get a complete list of files, whereas "*.*" and "*" are both necessary for a Tenex such as Maxc1. Alto FTP servers do not work as suitable hosts, since they do not support wild cards at all. Although hosts may be specified by a name or a network address, Brownie does not understand "3#200#" as being equivalent to "Maxc". Brownie treats any server name that starts with the string "maxc" as a Tenex.

Updating Directories

Normally, Brownie works by first enumerating files from both the source and destination hosts. The cursor blinks once for each file it enumerates. After the enumerations, Brownie then decides which files to transfer by comparing the write dates of files with the same name to determine if any files on the source [host]<directory> are newer than their counterparts on the destination [host]<directory>. This algorithm will be changed to use creation dates as soon as the file servers are brought up to the standard described in Ed Taft's memo on [MAXC]<AltoDocs>FileDates.press. If there are communication problems during any of these steps, Brownie automatically reopens connections after a two-minute wait.

Filenames are compared ignoring both version number and case. Subdirectories not specified as part of the command line are included in the file name for comparison purposes. This results in subdirectory structure being preserved for all transfer operations. In the sample directories below, we are updating [DestinationHost]<mumble>foo> from [SourceHost]<Mumble>Foo>. The file Bas>B.mesa will be transferred only if its write date is newer on SourceHost. The file a.mesa will be transferred regardless of its write date, because it is not on the DestinationHost. This will result in one version of a.mesa in the Foo>Bar> subdirectory and one in the Foo> subdirectory.

<pre>[DestinationHost] <Mumble>Foo> Bar> A.mesa!1 Bas> B.mesa!4 Baz></pre>	<pre>[SourceHost] <Mumble>Foo> a.mesa!2 Bas> B.mesa!3 Baz> a.bcd!1 b.bcd!2</pre>
--	---

Copying Directories

By specifying **CopyAll** on a command line, it is possible to instruct Brownie to copy all files, maintaining version numbers, from one directory to another. If any of the files exist on the destination directory, the new files will have higher version numbers.

Delver

Delver on a command line instructs Brownie to delete all but the latest version of every file in the destination directory using a telnet connection after the transfer is finished. This command will work only on an IFS. In addition, it is a requirement that the user's login grant the correct access permissions or that a connect password be specified.

Planned Enhancements

In the future Brownie may provide a mechanism for changing the contents of files as they are transferred. This feature is motivated by the desire to alter command files automatically to use the new host.

In addition, Brownie currently uses two connections when transferring files on the same host. It will probably be changed to use only a single connection when it can.

Problem Reporting

Any requests or problems concerning the use of Brownie should be sent to <SDSUPPORT> (8*823-2565). Please include pointers to the MESA.TYPESCRIPT and BROWNIE.FTPTRACE when submitting bug reports.

Sample Brownie.data

The format of Brownie.data has been changed since Brownie 1.0, mainly by the addition of new fields. However, old data files are compatible with the new format. In addition, Brownie is slightly more understanding about the placement of blanks between fields. Note that capitalization of commands is ignored. Despite these improvements, Brownie is still strict about the data file following the exact layout specified. The sample data file shown below is also stored on [IRIS]<BROWNIE>BROWNIE.DATA.

```

{brief}
TestFTP: {false}
Start: {20:00}
Stop: {2:00}
// logins follow
[Isis]<Malasky>()
[Iris]<guest>(guest)
[Maxc]<Malasky>(secret)
// These are commands
[Isis]<Mesa> (secret) ← [Maxc]<Mesa> () NoBuffering
[Isis]<OldMesa>4.2>() ← [Isis]<Mesa> () rename
[Isis]<MesaLib>3.0> () ← [Iris]<MesaLib>3.0>(foo) NoBuffering delver
[Iris]<Malasky> (secret) ← [Isis]<Malasky> () CopyAll

```

The first line controls how much information Brownie will log. The next three lines may occur in any order and are optional. *TestFTP* is for internal debugging, and may be **FALSE**. A *Start* time of **NOW** tells Brownie not to wait before running. Now is the default. The default *Stop* time is to run until finished. Times may be in any of the following formats: HH:MM, HHMM, H:MM, or HMM. The following line is a comment, but the leading **//** is required.

The next section is used by Brownie to log into the various hosts. A host is delimited by **[[**, a user name by **<>** and a password by **()**. Spaces are not allowed between fields of a login line. Logins are terminated by the second comment line. Only the two comment lines shown are allowed.

All the remaining lines in the data file are command lines specifying what updates are to be performed. Each command line is in this format:

```
[Destination]<To directory>(CP)SP+SP[Source]<From directory>(CP) TO CO DO CR ,
```

where

SP is a space,

CR is a carriage return,

CP is a connect password, possibly null,

TO is Transfer Options {**BufferFiles**, **NoBuffering**, **Rename**} or null (**BufferFiles**),

CO is Copy Options; **CopyAll** or null, and

DO is Delver Options; **Delver** or null.

No *****'s are permitted in any of the directory specifications. Subdirectories for **<To directory>** and **<From directory>** are permitted, but they are ignored by Maxc. The connect password is required only when the login name does not have the necessary access permissions. Nonetheless, the parentheses are required. Normally, a connect password will be specified for the destination [host] to store files, while one is usually not necessary to retrieve files.

ALTO HARDWARE CATALOG

Filed on [MAXC]<ALTODOCS>HardwareCatalog.Press

This catalog lists and briefly describes Alto hardware options and peripheral devices that may be attached to an Alto, along with their required interfaces. Devices included in this catalog have at least one prototype considered to work properly and to be reproducible from existing documentation. The standard Alto peripherals, display, mouse, keyboard, keyset, and Diablo model 31 disk have not been included. Each device has an entry of the following format:

ITEM NAME: Description

INTERFACE: Other options or modifications required.

SOFTWARE: What/Where is the appropriate code?

DOCUMENTATION: Where is it written down?

AVAILABILITY/SOURCE: Who did it and can I get one?

The entries are organized by hardware type: Alto Options and Chassis Mounted Interfaces, Alien Processors, Disks, Fabrication Equipment, Printers, Scanners, and Workstations.

This catalog is maintained and distributed by the *Alto User Support Coordinator*, Ron Cude. If you have questions or know of a device which you feel should be included, please provide the above information to Ron for inclusion in the next revision. Message <CUDE> or call Intelnet 8*823-2465.

ALTO OPTIONS AND CHASSIS MOUNTED INTERFACES

EXTENDED MEMORY: An Alto II option (standard on 8th. build and later machines) permitting memory to be increased from 64K to 256K in 64K chunks. Though the 16-bit addressing of the Alto prevents transparent use above 64K, data such as the display bit map may be placed there and code may be executed out of any 64K bank.

INTERFACE: Modifications to the Backplane, AIM, DIM, CRAM, Control, and Memory boards are required for pre-seventh build Altos. A MEAT module must be purchased to replace the old Terminator module.

SOFTWARE: Microcode AltoIIcode3XM.MB.

DOCUMENTATION: Alto Hardware Manual.

AVAILABILITY/SOURCE: Contact SPG for modification drawings and details.

EIA/RS-232C: A serial communications module supporting up to eight lines in either synchronous or asynchronous mode at 14 selectable transmission rates from 50 to 9600 baud.

INTERFACE: None.

SOFTWARE: STREAMS type package.

DOCUMENTATION: EIA Board Specification, memo from Shingo Arase, Aug. 31, 1977; Engineering drawings; "EIA Streams Package", memo from Rick Tiberi, Dec. 9, 1977.

AVAILABILITY/SOURCE: 24 were built to order by XEOS.

EIA/RS-232C, 300 BAUD: A simple serial communications module.

INTERFACE: None.

SOFTWARE: An assembly language driver and supporting microcode.

DOCUMENTATION: "Alto II 300 Baud Serial EIA Interface", memo from Gerald Justice, Feb. 17, 1977.

AVAILABILITY/SOURCE: Developed for special project by XEOS.

ORBIT: A four module interface designed to help the Alto convert character code/position information to a format suitable for a scanning laser printer.

INTERFACE: Alto II with ORBIT backplane.

DOCUMENTATION: "ORBIT General Description", Severo Ornstein, April 22, 1977; "Programmer's Guide to ORBIT, the ROS adapter, and the Dover Printer", Bob Sproull, June 15, 1977.

AVAILABILITY/SOURCE: Contact SPG.

ORBIT BACKPLANE: An Alto II backplane that has been modified to accept and operate the ORBIT interface.

INTERFACE: None.

DOCUMENTATION: Engineering drawings.

AVAILABILITY/SOURCE: Contact SPG.

TRIDENT DISK INTERFACE (TRICON): A disk controller for the Trident T80 or T300 model disks.

INTERFACE: None. May be some wiring changes????

DOCUMENTATION: "Trident Disk for the Alto" filed on [MAXC]<AltoDocs>Trident.Ears.

AVAILABILITY/SOURCE: Contact SPG.

TRICON MULTIPLEXER: Used to interface 2 to 8 Trident disks to the same Alto.

INTERFACE: Trident Disk Interface.

DOCUMENTATION: Engineering Drawings.

AVAILABILITY/SOURCE: Contact SPG.

WIRE-WRAP BOARDS, GENERAL PURPOSE: Four quadrants, each of which accepts eighteen 14/16 pin dip packages, separated by a longitudinal row of 92 pads and two horizontal rows of 27 pads that accept .300, .400, or .600 center dip packages (SSI, MSI, LSI).

INTERFACE: None.

DOCUMENTATION: PWB, GENERAL PURPOSE WIRE WRAP, Dwg. 596P61506.

AVAILABILITY/SOURCE: 40 were fabricated by XEOS Publishing Programs Department and are currently available.????

2K CONTROL MODULE: Replaced the previously standard Alto II 1K Control Module for applications requiring additional microcode such as Trident based file systems. It has been standard since 6th. build machines.

INTERFACE: None.

DOCUMENTATION: Alto Hardware Manual.

AVAILABILITY/SOURCE: Contact SPG.

ALIEN PROCESSORS/INTERFACES

XEROX SIGMA 3/7/9 INTERFACE: This line of processors has been interfaced to the Ethernet.

SOFTWARE: Alto and Sigma 3 programs to facilitate data transfer.

DOCUMENTATION: WRC Report X7703234, Mar. 77.

AVAILABILITY/SOURCE: Built for special project by WRC.

DACONICS VT-2 INTERFACE: The VT-2 (and soon VT-3????) has been attached to the Ethernet via the standard transceiver and modified to support the EFTP protocol.

SOFTWARE: Specially written for VT-2 and Alto II to support a Trident based filing system (but not TFS).
 DOCUMENTATION: Drawings of hardware interface.
 AVAILABILITY/SOURCE: Developed for special project by ASD.

XEROX 800 ETS, COMMUNICATING: A typewriter/cassette based word processing system.

INTERFACE: 300 baud EIA board.
 SOFTWARE: None?????
 DOCUMENTATION: None?????
 AVAILABILITY/SOURCE: Xerox

DISK DRIVES

DIABLO 44: Similar to the Diablo Model 31 except having an additional fixed disk, shorter seek times, higher transfer rate and sounds like a Boeing 747 taking off.

INTERFACE: Requires a different clock crystal on the Alto's Disk Controller board and a new cable.
 SOFTWARE: None. Executive treats it as a dual Model 31 Alto (fixed disk is drive 1).
 DOCUMENTATION: Alto Hardware Manual.
 AVAILABILITY/SOURCE: May be ordered from Diablo; cable fabricated by buyer.

TRIDENT T-80/T-300: Removable media drives with 80 megabyte and 300 megabyte capacities respectively and a 9.7 megabit transfer rate. They support the Trident and File Server systems.

INTERFACE: Trident Disk Interface (TriCon) and 2K Control Module.
 SOFTWARE: Interim File System, TFU utility subsystem, TRIEX maintenance subsystem, and TFS support package.
 DOCUMENTATION: "Trident Disk for the Alto", filed on <AltoDocs>Trident.Ears.
 AVAILABILITY/SOURCE: Century Data Systems, a Xerox company. Contact SPG for appropriate cables.

FABRICATION EQUIPMENT

PRO-LOG 92 PROM PROGRAMMER: A multi-personality PROM blower.

INTERFACE: Plugs into the Alto's HYTYPE port, special PROM's required for the PROM blower.
 SOFTWARE: [MAXC]<Alto>Pnew.Run.
 DOCUMENTATION: [MAXC]<AltoDocs>PromManual.Bravo/Press.
 AVAILABILITY/SOURCE: Pro-Log Corp. (Buyer must fabricate cable and blow the special PROM's).

PRINTERS

COLOR GRAPHICS: A 6500-based scanning laser, three color printer with a resolution of 200 bits/inch horizontal, 100 bits/inch vertical. Image width is limited to 6.4 inches, length to that of paper. Developed for use with color CRT's to provide hardcopy output.

INTERFACE: SLOT/3100 interface board.
 SOFTWARE: Basic software to process AIS files.
 DOCUMENTATION: Xerox Technical Bulletin 610P11514 and Advertising Bulletin 610P11259.
 AVAILABILITY/SOURCE: Xerox.

DOVER: A 7000-based variable rate scanning laser, high volume printer. A set of fonts has been developed to run at 384 scanlines/inch.

INTERFACE: ORBIT and an Alto II with an ORBIT backplane.

SOFTWARE: Spruce (server subsystem), Empress and others (user subsystems).

DOCUMENTATION: "Overview of the DOVER", John Ellenby, February 24, 1977.

AVAILABILITY/SOURCE: No longer being manufactured. Developed jointly by PARC and SPG. Some may be available from ASD????

FUJI XEROX 1660: A 660-based low cost, 200 bits/inch optical fiber printer.

INTERFACE: Connects to HYTYPE port.

SOFTWARE: Project specific software and microcode exist to convert text files to bands-type bitmaps and to transfer the bitmaps to the printer. WHOSE PROJECT????

DOCUMENTATION: Engineering drawings.

AVAILABILITY/SOURCE: Fuji Xerox markets the printer in Japan.

PIMLICO: A 6500-based variable rate scanning laser, three color printer.

INTERFACE: ORBIT and an Alto II with an ORBIT backplane.

SOFTWARE: Spruce (server subsystem), Empress, Color-Draw, Bravo 7.1X, and others (user subsystems).

DOCUMENTATION: Engineering drawings.

AVAILABILITY/SOURCE: No longer being manufactured. Developed jointly by PARC and SPG.

PRINTER MULTIPLEXER: A small black box device which allows a single server processor to operate up to four 9-wire interfaced printers. Two operational modes are provided: MANUAL, via a rotary switch; and AUTO, via software control from the server processor. No software is currently available to support the AUTO mode.

INTERFACE: ORBIT and an Alto II with an ORBIT backplane.

SOFTWARE: None required for MANUAL use, not available for AUTO use.

DOCUMENTATION: Engineering drawings.

AVAILABILITY/SOURCE: Contact SPG, four are currently available.

SEQUOIA: A 3100-based variable rate scanning laser, high quality, limited volume printer.

INTERFACE: ORBIT and an Alto II with an ORBIT backplane.

SOFTWARE: Spruce (server subsystem), Empress, and others (user subsystems).

DOCUMENTATION: Engineering drawings.

AVAILABILITY/SOURCE: Unknown at this time. Developed by XEOS.

SLOT/3100: A 3100-based scanning laser, limited volume printer.

INTERFACE: SLOT/3100 interface board.

SOFTWARE: Press.

DOCUMENTATION: Xerox Report X7902143, *SLOT Revisited*.

AVAILABILITY/SOURCE: A limited number were built.

TC-200: see TC-200 entry under SCANNERS.

VERSATEC D900A: An electrostatic printer.

INTERFACE: Standard Alto HYTYPE connector.

SOFTWARE: Vprint (.tty and Bravo/Gypsy files), Press (press files).

DOCUMENTATION: Alto Hardware Manual.

AVAILABILITY/SOURCE: Versatec (XEOS Publishing Programs Department has a few extra????).

SCANNERS

JOYCE-LOEBL SCAN-DIG 3: A selectable rate scanner (25/50/75/100 micron) for color or greytone material up to 10 by 10 inches.

INTERFACE: Specially built for Alto II.

SOFTWARE: Controls scanner and converts output to AIS format files for Alto and Trident file systems.

DOCUMENTATION: NSIL drawings of the interface.

AVAILABILITY/SOURCE: Built for special project.

TC-200: A variable rate scanner (94-240 points/inch) and Alto screen resolution printer. The supporting software creates/prints PRESS format files.

INTERFACE: Extensive modifications to the TC-200 and an Interface Box that attaches to the Alto via the HYTYPE connector.

SOFTWARE: Press for printing on an Alto I, modified Press and microcode for printing on an Alto II. New subsystem for scanning (unnamed).

DOCUMENTATION: None currently.

AVAILABILITY/SOURCE: Uncertain at this time.

WORKSTATIONS

REMOTE WORKSTATION: Permits placing an Alto workstation (minus keyset) and HYTYPE printer up to 1000 feet from the Alto.

INTERFACE: None.

SOFTWARE: None.

DOCUMENTATION: Engineering drawings.

AVAILABILITY/SOURCE: Built for special project.

VIDEO BUFFER BOX: Supports the connection of 5 displays to a single Alto.

INTERFACE: None

SOFTWARE: None.

DOCUMENTATION: Engineering drawings.

AVAILABILITY/SOURCE: 15 were built by Cybernex.

WORKSTATION MULTIPLEXER: A 4 by 4 switch that permits manual selection of an unused Alto from any workstation.

INTERFACE: None.

SOFTWARE: None.

DOCUMENTATION: Engineering drawings.

AVAILABILITY/SOURCE: May be ordered from Cybernex.

FUTURE TRENDS IN ELECTRONIC PUBLISHING

*Don Winter and Don Stewart
Xerox Electro-Optical Systems
Pasadena, California*

Electronic Publishing is not a speculative notion, it is not prediction, it is accomplished fact. The major question regarding the use of electronics and computer techniques in publishing is not whether, or when, or how, but how much, and how fast. Already, the vast majority of newspapers in this country is published electronically all the way from the keying of a story by the reporter through to composition and phototypesetting of composed galley. Electronic publishing has now spread to the weekly magazines, encompassing, in this case, full-page composition including *all* pictures, as well as electronic transmission for typesetting at remote printing facilities.

At the other end of the process, the Library of Congress no longer maintains a vast pre-printed inventory of its library cards from recent years, but rather prints cards electronically 'on demand'—on request from a specific user—using a laser xerographic printer and the magnetically stored MARC data base. The electronic printer, in this case, has simply been substituted for the former phototypesetter used to produce the masters from which the inventory was printed. The typeface and card layout produced by the electronic printing system are virtually indistinguishable from those used in the former card inventory.

This paper covers the present situation and future trends in electronic publishing, as the areas presently making use of electronic aids expand and grow together into a cohesive whole in which authorship, publishing, and 'reading' become intermingled, and in some cases indistinguishable, activities.

— 2 —

ELECTRONIC PUBLISHING TODAY

The publishing industry is presently in the throes of a profound technological revolution. New electronic tools make it possible to re-think and re-structure the editorial process. It is important to step back and look at the entire process from author to the final user in light of what is now possible that was not possible only a few years ago. We must consider how information is 'created', edited, formatted and disseminated.

INPUT:

Virtually everyone who has had a chance to use a good video terminal to write copy will agree that it is a far better device than a typewriter. The keyboard is faster. It is easy to correct mistakes or to change your mind and re-work copy. It is easier to move copy around—especially since you always have a clean version of your manuscript to work with. It is possible to save the current version of a section of the text while you work on an "improved" one. If the new version is, indeed, an improvement, you can kill the original. If not, you can go back to the original. And, there are facilities such as string searches and string replaces which help the author to find words and phrases he wishes to check for accuracy or consistency.

Where it is possible for an author to work directly on a video terminal, the industry has found that he feels that he can do a better job, and that the copy which he writes should never have to be re-typed or re-proofread for new typographic errors before it is put into final form to be distributed to the 'customer'.

Where it is not feasible for the author to work directly on a video terminal, it is sometimes possible to 'capture' his original work via an OCR reader or (in some situations) an interface which will accept input from a simple word processor. Most commonly, however, it makes most sense to 'capture' the material the next time it is re-keyboarded. As a general rule, the greatest benefits can be derived from the new technology if the data can be 'captured' as early in the editorial/production process as possible and advantage taken of electronic tools from that point forward.

EDITING

It has been the newspaper industry which has seen the first wide-scale use of video terminals as writing and editing tools. Within approximately a half-dozen years the industry has progressed to the point where the video terminal has replaced pencil and paper as the dominant working tool for editors. Over this period of time the companies which supply electronic systems to this industry have improved terminals and terminal functions to make the terminal an effective tool for the working editor.

— 3 —

In the word-processing industry, on the other hand, the 'editor' has generally continued to work with hard-copy 'print-outs' rather than working directly on the video terminal himself. The video terminal has been introduced for use by secretaries who implement the changes marked on paper by transcribing them to the terminal screen, then asking for a new printed copy to hand back to the 'editor'. This requires an extra person in the process and will naturally take longer than if the editor were to make the changes himself. However, if the 'editors' do not edit much copy (as would be true in an office environment) it may be cheaper than giving each 'editor' his own terminal. And, if the editor does not know how to type, it may be the only realistic solution.

For many of the situations found in the commercial and scholarly publishing world, it will be most effective to follow the 'newspaper' example and have editors working directly on video terminals. However, there are situations in which it will remain more desirable to have editors work with pen or pencil on paper and to have someone else transcribe the changes, as in the word processing example. A good terminal will generally be a better working tool for the editor if there are a moderate number of changes to be made. He will find the fact that he can verify what he has done as he is working rather than wait for it to be implemented by someone else and returned to him to be a major advantage. If he has only very light changes to make, he may find that he can flip through paper faster than working on a screen—although even here he may prefer the screen because of the immediate feedback and having him work directly on the screen may save the extra labor cost of having someone else to transcribe his changes.

REVISIONS

One of the real advantages of the new electronic systems is the ease with which previously-input material may be revised. Only the changes need be keyed and verified. And, the new version may appear in an entirely different format and/or be produced on a different type of output 'printer' than was the previous version.

MULTIPLE USE OF DATA

A related advantage is the opportunity for multiple uses of the same information. A few of the possibilities which come to mind include the direct transcriptions of abstracts and key words for secondary publishing and indexing services, as well as the capture of the full-text of articles, monographs, and books themselves for use in electronic full-text form or as the data-base for electronic 'demand publishing' operations. These possibilities may barely scratch the surface. Some of the publishing companies who moved into electronic systems early have become quite creative in exploiting the information which they have stored electronically.

TEXT ANALYSIS

One of the potential benefits of an electronic text processing system is the use of the computer to check and analyze text which is stored in the system. Relatively few companies have really exploited the possibilities yet, but there is increasing interest in the concept. Some of the possibilities include:

- Checking the words in a document against a relatively large dictionary to check for spelling accuracy. Some systems flag all of the words in the document which are not in the dictionary as words which are possibly mis-spelled. Some will even change the spelling of frequently mis-spelled words (with an indication to the editor that the change has been made).

- Checking a document for the occurrence of certain words which are considered appropriate or in-appropriate for certain reading levels.

- Applying a standard test for reading difficulty.

There are others as well — probably including some things that no-one has yet thought of. This sort of analysis is particularly useful for companies which are involved in educational publishing, but they may be useful in other contexts as well. One of the best-known uses of a dictionary to check spelling is in the work which International Computaprint processes for the Patent Office, and one of the first companies to incorporate tests for reading difficulty in its text processing system is an insurance company.

FORMATTING

The better text processing systems now provide very powerful capabilities for formatting output copy. These include high quality hyphenation/justification routines to format copy into lines and page make-up facilities to format pages. In a number of cases it is possible to accomplish this without requiring that the user imbed esoteric commands in the text file. This, again reflects the fact that much of what has been developed over the past half-dozen years especially is intended for use by a non-technical writer or editor.

OUTPUT

The new electronic devices provide powerful tools for writing, editing, updating and formatting material. They also bring about a situation in which the mechanical "production" aspects of typesetting and even printing become a simple extension of the editorial process. The hardcopy output device may be a daisy-wheel printer, an electronic printer such as the Xerox 9700—a modified version of which forms the basis of the Library of Congress card printing system mentioned earlier—or a phototypesetter. All function as fancy computer output printers and it is not all that material to the electronic system (or to the user) which one is used.

ELECTRONIC PAGE MAKE-UP AND DIGITIZED GRAPHICS:

Initially it has been important to concentrate on perfecting the use of electronic tools for input and editing text. However, many publishing organizations will soon be able to move into electronic page make-up for a significant portion of their work. Journal publishers could assemble pages for their periodical production by entering the page layout or page dummy information into the electronic system, then using a graphic display screen to manipulate the text elements to appear on the page. Publishers may be able to handle book work as well by using some combination of automatic page make-up program and interactive graphic display.

As publishing organizations gain more experience in using electronic systems and as the page make-up functions available in the systems improve, they should be able to make more extensive use of electronic page make-up for an even larger proportion of their product. Beyond this, it is anticipated that they would be able to move into storing, manipulating and outputting graphics (line drawings, half-tones and even color pictures) in digital form so that they will be able to produce complete pages with all elements (text and graphics) in position.

As alluded to earlier in this paper, *U.S. News and World Report* has been producing all editorial pages in this fashion since the summer of 1977. All line art, half-tones and screen tint backgrounds (including colored screens and spot color) are digitized and stored in the text processing computer (an Atex system). Pages are set on a Information International VideoComp CRT typesetter at the *U.S. News* offices for proof purposes. Pages to be printed are transmitted to other VideoComps located at remote printing plants where the complete pages are "typeset" on film which is used to make printing plates. *U.S. News* does not, in early 1979 use interactive graphic display tubes to speed the process of assembling the elements to appear on a page—but this should come before the middle of 1980. Nor does it yet digitize and transmit color separations—but this will come within the next few months.

In early 1979, because of the cost of the equipment used to digitize pictures and artwork, and the cost of the digital storage required to accommodate digitized graphics, the *U.S. News* approach makes sense only in situations in which there is a high value to be gained by transmitting complete pages in digital form. However, there is little doubt but that as the technology improves an increasing number of firms will find that it pays to handle graphics in digitized form—especially if they expect to be able to output complete pages directly to an electronic printer or a laser plate-maker.

A DIGITAL WORLD

In a longer view, electronic publishing systems do more than provide a better means of producing "traditional" published materials. They also provide a bridge into a world in which information is stored and transported in digital form rather than as images on paper. Digitally stored information may be transferred to paper when, where, and as needed. It may never be transferred to paper at all. The "publisher of the future" may think of himself as being in the information business or the information/entertainment business rather than the book or magazine publishing business.

THE PROGRESS OF THE REVOLUTION

People have been working with computer-based text processing and typesetting systems for almost twenty years, and have been using computer systems to assist in the editorial process for books and magazines at least since *Time inc.* installed its system in the mid-60's. However, the real impact of the revolution has been surprisingly recent—just since the commercial use of video terminals for text editing. The move to electronic publishing systems passed a major milestone with the first use of video terminals by the *Associated Press* and the *United Press International* in 1970. But the real watershed event was the installation of the first electronic newsroom system at the *Detroit News* in 1972. In just six years since that time, virtually every newspaper of any size in this country either has installed such a system or plans to do so. Some are already on a *second generation* system!

From newspapers, the revolution has spread to magazines, government agencies, "in-plant" documentation centers, and even some book publishers. It seems likely that no publisher, save perhaps the very small 'garage' operations, will be immune to the changes resulting from the Electronic Publishing revolution.

FUTURE TRENDS

A number of possible future developments and trends in the on-going Electronic Publishing revolution has been alluded to already in this paper, along with the discussion of those areas of the revolution which have already started, and in some cases, reached a degree of maturity.

EDITORIAL SYSTEMS

Editorial systems—those used for keying, editing, revisions, and text analysis—represent pretty much a mature technology already, except in the instance of text analysis software. The improvements to be expected in the next few years will be limited to the introduction of partial page or single line display terminals for

repro typist input of hard copy, and perhaps the increasing use of OCR devices for hardcopy input where the volume warrants. In other respects, editorial systems will remain very much like the video terminal systems already in use in newsrooms, except that these systems will become steadily cheaper and thus more widespread.

COMPOSITION SYSTEMS

Composition systems—those used for formatting, page makeup, and the inclusion of digitized graphics—will see great advances in technology and functional richness in the next several years. Systems capable of performing each of the necessary functions have already been demonstrated, but integrated systems remain deep in the research laboratories, as yet.

Improvements to come include systems which are capable of dealing with form and content simultaneously, systems with (perhaps interactive) hyphenation and justification routines which are adaptive as the page layout changes, and display the results immediately to the user, systems capable of the interactive creation and scaling of synthetic line art, and systems capable of the inclusion of all of these, together with scanned-in art and pictures, in interactive page makeup. Increasingly, such systems will also possess the capability to deal with color specifications for text and synthetic graphics and with digital color separations for scanned-in graphics and pictures.

OUTPUT AND PRINTING SYSTEMS

One of the more recent developments in Electronic Publishing has been the introduction of laser xerographic output printers capable of producing reasonable facsimiles of graphic arts quality typefaces, and of including these along with line graphics and perhaps halftones on fully made-up pages. These devices are capable of producing the entire press run of limited-distribution publications directly from the electronic memory to the final printed page. It is one of these devices—a modified version of the Xerox 9700—which forms the basis for the Library of Congress' card printing system mentioned earlier.

Electronic printing using laser xerography has also been applied to the *color* xerographic marking engine. This type of electronic publishing output has applications even at the high-quality end of the spectrum for publications in which the final product is to be typeset and printed on gravure or offset color presses. The electronic color printer allows prepress proofing of color masters without extensive (and messy) photographic techniques, which produce colors that are at best approximations to those which are obtained from a printing press, since they (unlike the xerographic ones) are formed in a different way.

The addition of communications links to the electronic publishing system will also allow the use of remotely-located electronic printers for the rapid distribution of finished documents.

SECONDARY USES OF ELECTRONICALLY PUBLISHED DATA

Computerized literature searching services have been available to librarians and information specialists for some time now. In some cases, the abstracts and index terms used in the data bases provided on these systems have been obtained in digital form from magnetic tapes provided by the primary publisher as a spin-off from his computerized composition activities. In many more cases, however, the abstracts and index terms have had to be manually re-keyed from printed versions of the primary material.

This situation is likely to change rapidly as more and more journals and monographs are composed electronically, and in many cases the transfer from primary to secondary publisher could take place electronically over communications links, rather than by the physical transfer of magnetic tapes. The full-text of journal articles and monographs may also be captured as 'full-text files' by these means, with the co-operation of primary publishers, possibly in an integrated system if a standard format can be adopted.

The technological development of trillion-bit computer memories, which can store entire libraries (or at least, whole sections of libraries), better access methods, the growing network of interconnected computer-communication networks, the local usage of personal computers with CRT displays rather than 'dumb' hardcopy terminals for reading, optical disks for local storage, and speedy hardcopy printer systems, together point to the real possibility of a national or international system of full-text files. The functions of libraries could change drastically as they move from their traditional role as store houses of collections of materials to becoming dynamic nodes in information networks

ELECTRONIC JOURNALS

Computerized message systems are already in use by some groups of workers engaged in closely related activities in geographically dispersed areas. These systems provide for asynchronous communications between or among any subset of the group, and allow the easy transmission of data which might be difficult to transfer by means of voice communications. The computer's ability to retain a permanent record of these messages allows one to compile a sequence of interactions on a particular subject, which is, after all, the original meaning of journal.

The advent of full-text files would allow the publication of completely electronic journals. The *meaning* of publishing, and indeed of authorship, could change in substantive ways. For example, the legal moment of publishing could be either the time the author enters his text into the system, perhaps by means of his personal computer, or the time when it is first called up for reading by a user. For authors, the satisfactions and prestige of “seeing their work in print” could be largely diminished, so that motivations to publish might be altered. The role of ‘editors’ also could change considerably, and the protocols for refereeing professional papers would need to be restructured for the all-electronic medium.

The hardcopy replication of digitally stored text at widely dispersed points of use and sale—a form of demand publishing—will be a natural adjunct of full-text files and electronic journals.

DEMAND PUBLISHING—PAPER IS NOT DEAD

As mentioned earlier, library cards for materials published in the United States since 1968 are currently being printed electronically directly from the computerized database. Although this is taking place in a centralized location, it is easy to imagine developments leading to the use of similar (but cheaper) printers connected to one’s personal computer for local printout, rather than distribution through the mails. Eventually, as the files created during computer composition and typesetting of publications come to be collected and stored, one’s local printer might be used for creating a paper copy of a stored publication ‘on-demand’.

SUMMATION

These are just some of the trends which have been forming in the Electronic Publishing Revolution within the last few years, and which can be expected to come to fruition and into widespread use in the publishing industry in the next several years.

MICROGRAPHICS AND PERSONAL COMPUTERS IN INFORMATION SCIENCE IN 2001

Don Winter and Don Stewart
Xerox Electro-Optical Systems
Pasadena, California, 91107, USA

A growing revolution is taking place in the ways in which information is acquired, used, manipulated, created, and stored. This paper discusses the impacts which are projected for micrographics and personal computers in the field of information science by the year 2001. It covers the rapid emergence of new and improved technologies, both in information science and in the world at large, which are leading us into what Daniel Bell has called the *Post-Industrial Society*.

Before proceeding with any discussion of the impacts of the new breed of *personal* computers and of the relatively old, but still emerging, micrographics it would be well to describe what we will mean by these terms as they apply to the years remaining in the twentieth century.

A **Personal Computer** provides a user with his own private processing space and storage space, inaccessible to others without his active cooperation. In this respect, personal computers differ from central computers in that they have local file storage and from computer terminals in that they have local, private, processing space. Even so-called intelligent terminals generally require access to a central computer for program loading, file access or both. By the turn of the century, personal computers will not be stand-alone devices, but will have virtually instantaneous access to a multitude of shared-resources over communications links. The user's (private) local files will be accessible to him alone, but much information will be stored in public files, accessible to all, or in 'group' files, accessible only to members of formal (or perhaps informal) groups.

— 2 —

Micrographics is the art of transforming information into reduced imaginal form for manipulation and storage. Traditionally, this has been performed by photographic means, but holography, as well as facsimile techniques using optical disks, must also be considered for the future. With the emergence of computer-output-microfilm, computer-indexed and accessed microform data-bases and the forthcoming optical disks, the merger of computers and micrographics has begun. Note that an optical disk is not a synonym for a video disk, despite the fact that one of the video disk approaches uses optical reading techniques.

This paper is concerned with the ways in which personal computers and micrographics will be used for information storage, access, acquisition, use, manipulation and creation. The applications of personal computers and micrographics will include the areas of:

- o libraries as a means of access to information, rather than as repositories of paper;
- o electronic journals and other information sources;
- o demand publishing of paper copies of articles, books, reports, and other information sources; and,
- o imaginal information.

The reasons for the inclusion of personal computers and micrographics in the overall information system of the future are not hard to discern. Microform represents a very economical form for the storage of imaginal information and as a replacement for paper in locally stored manual indices etc., and makes a good data base for demand publishing applications, although it may eventually be supplanted by completely digital storage of current information and publications.

Personal computers are already starting to be used in all facets of the overall information system, with the exception of mass central storage. Personal computers are especially adaptable to the creation, acquisition and manipulation of information, such as the printed version of this paper, and have been known to make drastic changes in the habits of authors. Self-plagiarism, for instance, becomes ridiculously easy, and the pain of extensive rewrites is considerably reduced, perhaps to the point of encouraging an excessive number of iterations of a document.

Let us consider these areas of applications in turn, and then go on to the world of 2001 and how to achieve it.

LIBRARIES AS A MEANS OF ACCESS TO INFORMATION RATHER THAN AS REPOSITORIES OF PAPER

One hears and reads a lot these days about 'networking' for libraries, and its potential for improving a library's access to materials not in its collection. At present, a 'network' in this sense seems to mean a set of telephone lines or other communications links providing one with access to a bibliographic data base located on a large central computer. Of course, a library's personal operations, such as computerized shelf-lists, circulation systems, and the like, are usually located locally to the library, but these operations are not often directly connected to the network access mechanism, as yet.

Libraries have traditionally viewed their role in society as amassing a collection of books and periodicals in order to maintain a store of the accumulated knowledge of the world. It has begun to be realized in recent years, however, that such a collection, however comprehensive, is of little value if the information therein cannot be conveniently accessed when, or where, it is needed. The role of the library is thus being reassessed as that of providing a means of access to information. The increasing costs of labor and the shelving and facilities required to store the collection, coupled with the slow rise in library operating and capital budgets has squeezed acquisitions budgets so that libraries are also economically unable to collect as much.

Computer based bibliographic services and abstracting and indexing services have become increasingly more effective over the past decade, so that advanced libraries now can search automatically through many large and comprehensive data bases, including (largely current) periodical and report data bases. Though there has been little pressure thus far—except in the legal field—to develop the digitized storage of complete texts and graphics of articles and books—known as 'full-text files'—rather than simply abstracts, the delays between finding abstracts and retrieving the complete information content have become a significant factor in information services. It may take one half hour to find a desired reference and weeks to get a copy mailed from a distant library.

The technological development of trillion-bit computer memories, which can store entire libraries, better access methods, the growing network of interconnected computer-communication networks, the local usage of personal computers with CRT displays rather than 'dumb' hardcopy terminals for reading, optical disks for local storage, and speedy hardcopy printer systems, together point to the real possibility of a national or international system of full-text files. The fact that much current

information is composed and typeset through computer techniques makes such information available for filing and storage, possibly in an integrated system if a standard format can be adopted. The functions of libraries could change drastically as they move from their traditional role as store houses of collections of materials to becoming dynamic nodes in information networks

ELECTRONIC JOURNALS

Computerized message systems are already in use by some groups of workers engaged in closely related activities in geographically dispersed areas. These systems provide for asynchronous communications between or among any subset of the group, and allow the easy transmission of data which might be difficult to transfer by means of voice communications. The computer's ability to retain a permanent record of these messages allows one to compile a sequence of interactions on a particular subject, which is, after all, the original meaning of journal.

The advent of full-text files would allow the publication of completely electronic journals. The *meaning* of publishing, and indeed of authorship, could change in substantive ways. For example, the legal moment of publishing could be either the time the author enters his text into the system, perhaps by means of his personal computer, or the time when it is first called up for reading by a user. For authors, the satisfactions and prestige of "seeing their work in print" could be largely diminished, so that motivations to publish might be altered. The role of 'editors' also could change considerably, and the protocols for refereeing professional papers would need to be restructured for the all-electronic medium.

The hardcopy replication of digitally stored text at widely dispersed points of use and sale—a form of demand publishing—will be a natural adjunct of full-text files and electronic journals.

DEMAND PUBLISHING—PAPER IS NOT DEAD

Library cards for materials published in the United States since 1968 are currently being printed electronically directly from the computerized database. Although this is taking place in a centralized location, it is easy to imagine developments leading to the use of similar (but cheaper) printers connected to one's personal computer for local printout, rather than distribution through the mails. Eventually, as the files created during computer composition and typesetting of publications come to be collected and stored, one's local printer might be used for creating a paper copy of a stored publication 'on-demand'.

Digital storage and electronic publishing techniques are not, of course, *requirements* for publishing on demand. Alternative demand publishing methods include the substitution of an automated regional repository, duplicating library, or demand publishing service for conventional publishing, library service, or interlibrary loan service. The storage may be in any form available, although we are concerned here mainly with the alternatives of microform and computer storage. Demand publishing techniques would allow the individual user of information to obtain only those items of interest to him, such as individual journal articles, thus enabling him to establish a comprehensive personal library for a price comparable to that of a small number of journal subscriptions.

Demand publishing allows easy retrospective acquisition of material needed when a user's interests change, which is presently inconvenient without extensive photocopying. Retrospective material would almost certainly be stored in microform, due to the prohibitively high cost of backfile conversion. Microform storage, of course, somewhat restricts the prospects of instantaneous access to information over a communications network. Mechanical and opto-mechanical devices are less amenable to automation and data transfer than electronic and electro-optical devices.

IMAGINAL INFORMATION—THE FUTURE ROLE(S) OF MICROGRAPHICS

Microform, today, is by far the most economical medium for the storage of imaginal information. It was long believed that microform was *the* solution to library storage problems, and to a certain extent it has been. However, the problems of indexing, the difficulty of retrieval, along with the lack of really satisfactory microform display-readers, the lack of integration between hardware and software producers, and the lack of standardization, have prevented microform from becoming the success that had been predicted. Moreover, clear reader preference for hardcopy over microform displays (except for quick look-up of facts and reference numbers) points to the role of microform as being chiefly in the area of back-file reproduction rather than current information, except in the areas of ephemeral or frequently updated current information, such as telephone numbers, library catalogs and parts catalogs. Even these may eventually be replaced by computer files, as digital storage and access costs continue to drop. Thus, technologically, microform appears as an interim solution to dissemination of catalogs and indices and storage of back files and full graphics until microform is overtaken by the development of digitized full-text files and network access to them.

The advent of wide-band communications networks permits microform to be utilized as a central data-base of imaginal information, to be accessed under

computer control. It is not clear at present when digitized storage of imaginal information will become more cost-effective than microform under these circumstances. The advent of optical disks, predicted to be on the market by the mid-1980s, should clarify this area somewhat. But no matter what happens, the imaginal information will be perceived by the user as being supplied by the computer, and not as a piece of film in a viewer.

THE WORLD OF INFORMATION IN 2001

What, then, are we expecting to see in the information world in 2001 as far as micrographics and personal computers and their respective contexts are concerned?

As stated earlier, microform, in its conventional sense, will likely be restricted to the storage of backfile and of high quality graphics. Even this is likely to be under computer control for retrieval purposes by the turn of the century. Non-conventional forms of micrographics will be indistinguishable from other pieces of computer systems, at least as far as the user is concerned. Micrographics as a whole will thus be considered to be subsumed into the overall context of an *information system* by 2001.

An information system consists of three main classes of people and organizations—the producers, distributors, and users of information—which sometimes overlap and are often indistinguishable from one another. The producers of information include authors and publishers; the distributors of information include libraries, networks, journal subscriptions, bookstores, literature-searching services, and information archives; the users of information include libraries, information specialists, and the individuals requiring the information, who may also be authors.

By 2001, the overall electronic information system will encompass every facet of the information communication process from authorship to end use of the information. In this overall electronic information system, the producers of information include word processing and document creation systems and electronic publishing systems; the distributors of information include the mass storage systems and electronic full-text file access over a network, and the users of information include local storage and local use on an office terminal using either individual or shared storage and processing. The systems used at the local level for information use and for information creation can obviously be identical, or very similar systems—what we are defining here as *personal computers*. The publishing systems may be very similar to, or expanded versions of, the document creation systems. Only the mass storage systems, and network access to their contents can not be considered directly as *personal computers*.

— 7 —

We are talking, then, about a world in which every information producer and user has his own personal computer for information generation and use, which is attached by means of communications links to one or more resource-sharing networks for information transmission, public storage, and retrieval. The geographical location of any given item of information will be completely transparent to the user, except, perhaps, for its speed of transmission to him.

HOW WILL WE GET THERE FROM HERE?

The nuclei of these personal computer/resource-sharing network systems exist today, both within the research laboratories of private organizations, and increasingly in government and the business world, perhaps as market probes.

There are strong pressures within the library world, at least within the United States, for the development of a comprehensive, integrated library system for access to the Library of Congress MARC data-base and to the nation-wide locator list for inter-library loan purposes. It is envisioned that the locator list, at least, may not be a single centralized file, but may have some geographical hierarchy to it, preferably one which is transparent to the user.

In terms of more generalized resource-sharing information networks, the existing nuclei within industry will doubtless be expanded to the full extent of the geographical spread of each organization. *Interconnection* of such systems remains problematical at best, except for electronic mail or messaging purposes. However, some level of standardization of formats will probably occur, to facilitate access of these systems to public data bases. This level of interconnection may well be all that is required to allow these systems to reach their full-potential for use in the world of information and information science.

As we develop the uses of micrographics and personal computers, we must be careful to keep the *system* aspects in mind, to identify the interfaces between separate portions of the system, and to determine where possible format problems, as well as both software and hardware incompatibilities may occur. In this way, the separately developed 'independent systems' can easily become the 'modules' of the overall information system when technological developments and reduced software, hardware, and communications costs permit its formation. Present trends suggest that this is likely to happen several years before 2001.

REFERENCES**A Technology Assessment of Advances in Scientific and Technical Information Services**

Don Winter, Don Stewart, and Nilo Lindgren

Xerox Electro-Optical Systems, April 1977

NTIS Accession Number: PB-265839

System Concepts for Network Use of Full-Text Files

Don Winter, Nilo Lindgren, Judy Kaye, and Don Stewart

Xerox Electro-Optical Systems, May 1977

NTIS Accession Number: PB-267206

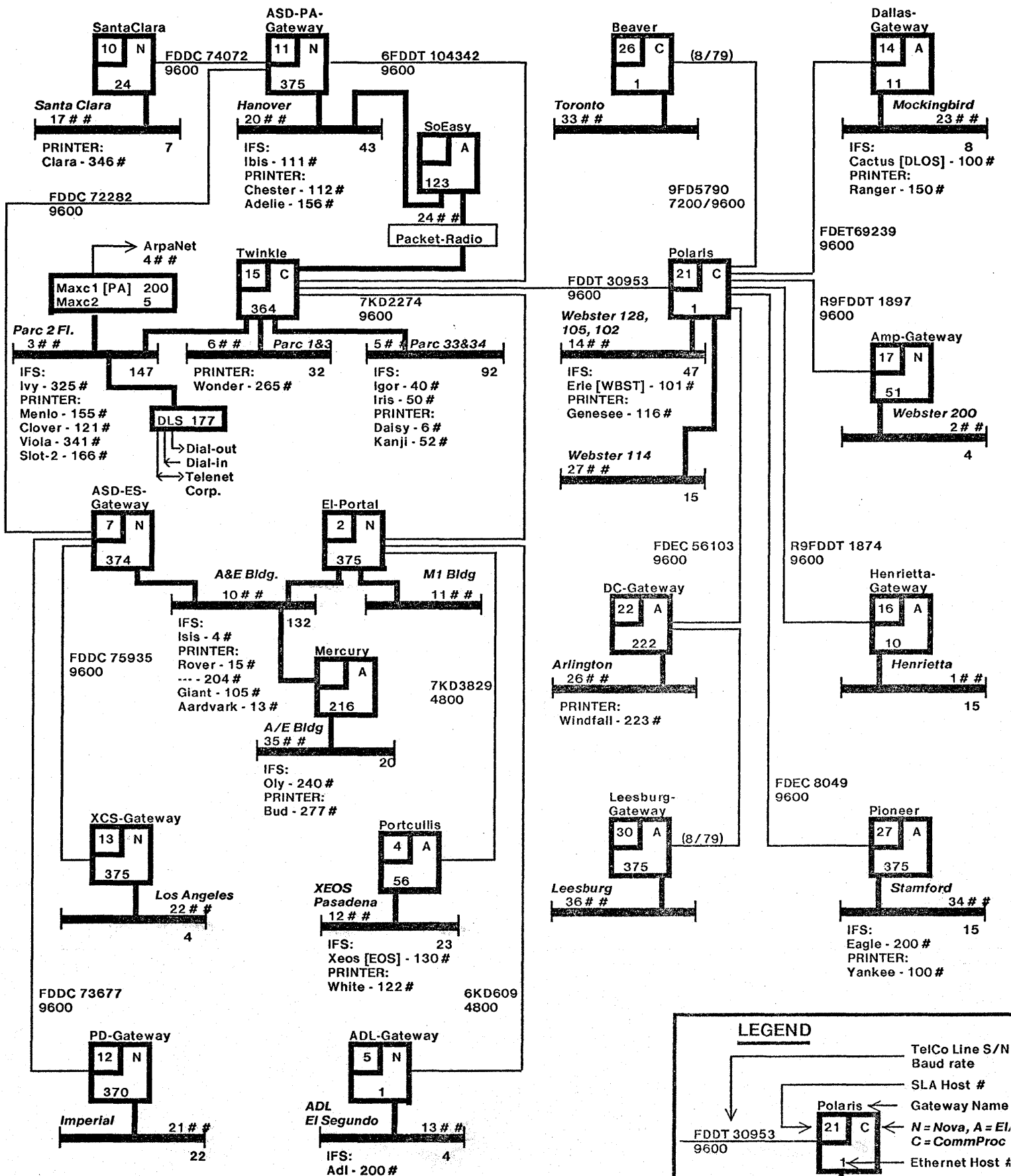
Electronic Printing: Another Facet to the Information Processing Revolution

Jonathan Seybold

The Seybold Report

Vol. 8, No. 5, November 13, 1978

The material in this paper is based largely on work performed for the National Science Foundation on contracts NSF-C1018 and NSF-C-05637.



**ALTO NATIONAL NETWORK
NETWORK TOPOLOGY**
August, 1979

LEGEND

- TelCo Line S/N
- Baud rate
- SLA Host #
- Gateway Name
- N = Nova, A = El, C = CommProc
- Ethernet Host #
- Location
- Network #
- Hosts on Net

[WBST] => Mail Registry

XEROX

Whole ALTO World Newsletter

For Xerox internal use only
Published and edited by Ron Cude
Message <CUDE> or call Intelnet 8*823-2465

TECHNOLOGY AND TOOLS FOR THE FUTURE

October 1, 1979

SPECIAL ANNOUNCEMENTS

IT'S TIME FOR ANOTHER MEETING

The date for our next meeting is Tuesday October 9, 1979. The meeting will be held at the TLC Best Western Hotel, 460 Totten Pond Road, Waltham, Mass., 617-890-7800. Darwin Newton, our host, has provided a list of other hotels in the area:

Holiday Inn Waltham, Totten Pond Rd., Waltham, 617-890-3000 (adjacent to the Best Western)
Waltham Motor Inn, 385 Winter St., Waltham, 617-890-2800
Sheraton Lexington, 727 Marrett Rd., Lexington, 617-862-8700
Marriott Hotel, 2345 Commonwealth Ave, Newton, 617-969-1000
Holiday Inn, 399 Grove St., Newton (Riverside), 617-969-5300
Wellesley Motor Inn, Rt. 9, Wellesley, 617-235-8555
Howard Johnson, 300 Washington St., Newton, 617-969-3010 (over turnpike)
Boston Sheraton Hotel, 39 Dolton St., Boston, 617-236-2000 (downtown)
Copley Plaza Hotel, Copley Square, Boston, 617-267-5300
Colannade Hotel, 120 Huntington Ave., Boston, 617-261-2800

WRC will have a series of demonstrations available Wednesday afternoon, October 10, for those who would like to see what's new at Webster. Make your plans now to attend. Please message or call Ron Cude to reserve a place at the meeting.

WHAT WILL WE CALL THE WAW???

Now that *DO's* will be augmenting, and at some point in time outnumbering the *Alto's* that are key to our **WHOLE ALTO WORLD**, the time has come to seriously think about changing our name to reflect more than just the **Wonderful World of the Alto**. So far, I've only received one idea (thanks, Ted). Let's give this some thought and try to come up with something that's both descriptive of our environment and, hopefully, clever and catchy.

GENERAL NOTES

NEW NETWORK DIAGRAM

New versions of the *Alto* Network Map and Network Topology Diagram have been stored on [MAXC]<AltoDocs>. Their names have been changed. They are now stored as **NetGeography.Press** and **NetTopology.Press** respectively. Copies can be found on pages 5 and 6.

WHOLE ALTO WORLD NEWSLETTER**THE EOS REGISTRY IS ON THE AIR**

The great mailbox switchover has been virtually completed as far as the people at XEOS, Pasadena are concerned. They are now all accessed at the EOS registry rather than on MAXC. This means that you can now address messages directly to the person of your choice rather than through <LizBond> as has been the case for so many of us in the past. Messages specifically intended for <LizBond.EOS>, <DonStewart.EOS>, <Hains.EOS>, <Pellar.EOS>, etc. should continue to be addressed to them by name, but now at the EOS registry.

If you do not know exactly whom you wish to address in Pasadena, send the message to <Lansford.EOS> for matters related to programs generated and/or modified at XEOS, <Hains.EOS> for font tools, and <DonWinter.EOS> for matters related to system administration, the [XEOS] IFS, etc. These people will let you know the appropriate addressee, and if in a sufficiently benevolent mood, may even forward the message.

For a list of those whom you can message at XEOS, see [XEOS]<Secretary>AllEOS.di. The MAXC distribution lists have been fixed up to reflect the focal points described in the previous paragraph.

Submitted by *Don Winter*, XEOS

OVERWRITING FILES ON IFS

Here's a hack that will stop your IFS account from keeping versions of things that you store so that you will no longer have to go back and do:

```
@Delete *,
@@Keep 1
```

Here is the kludge. IFS recognizes three meta-version numbers: L (*lowest*), H (*highest*) and N (*next*). Hence, if you do:

```
*Store (local file) foo.bar (as remote file)foo.bar!H
```

or on the command line:

```
>FTP Oly Store/s foo.bar foo.bar!H
```

you will overwrite the highest version of foo.bar. This even works when there is no current version of foo.bar stored on IFS.

Submitted by *Leo Nikora*, ASD

EXPANDED LAUREL MAIL SERVICE FROM GSD

Effective immediately, ESMail is expanding its Laurel hardcopy service for non-Laurel users. This added service will enable Alto users located virtually anywhere to send messages through ESMail to the majority of Xerox locations in the greater Los Angeles area.

Messages being sent to any of these locations should be addressed to EMPLOYEE NAME LOCATION CODE<ESMail>. For example, a message sent to John Doe at the Western Region Headquarters in Santa Ana should be addressed:

```
John Doe SAWE<ESMail>
```

Messages received by ESMail for these locations will be delivered through the Mail Systems Courier Network. Delivery of messages will have a maximum of an overnight turnaround.

WHOLE ALTO WORLD NEWSLETTER

Please direct any questions or comments regarding this service to Joe Alhanati at <ESMail>, El Segundo Mail Stop A1-50, or INTELNET 8*823-1554.

ADDED LOCATIONS TO BE SERVICED

LAWE..... Century City West Branch (2029 Century Park East, Los Angeles)
 LACE L.A. Central Branch (700 S. Flower St., Los Angeles)
 LASO L. A. South Branch (500 Carson Plaza Drive, Carson)
 ORGE..... Orange Branch (One City Blvd. West, Orange)
 SAFE..... San Fernando Branch (5901 DeSoto Ave., Woodland Hills)
 PARO XRO (125 N. Vinedo, Pasadena)
 COMP NDC (660 W. Artesia Blvd., Compton)
 POPC..... Pomona Printed Circuits (800 E. Bonita Ave., Pomona)
 SAWE Western Region Headquarters (2200 E. McFadden, Santa Ana)
 EMBR..... El Monte Branch (9350 Flair Dr., El Monte)
 LARC LARDC (13635 Freeway Dr., Santa Fe Springs)
 IRMC Irvine (18691 Jamboree Rd., Irvine)
 LXRC XRC (3255 Wilshire Bl., Los Angeles)
 IVXRC XRC (17781 Mitchell Ave., Irvine)
 LADC..... XDC (9200 Sunset Bl., Los Angeles)
 TAXT XTEN (21731 Ventura Bl., Woodland Hills)
 INXRC XRC (10926 S. La Cienega Bl., Inglewood)
 LACS XCS (5310 Beethoven St., Los Angeles)

MARKET PLACE

MARKET PLACE provides a forum for *Alto* users to make offerings and requests for *Alto* related hardware and software. To place an "ad" or offering, send the text to Ron Cude in El Segundo, message <CUDE>, or phone Intelnet 8*823-2465.

HARDWARE CATALOG UPDATE

The *Alto Hardware Catalog* is still being updated. (Primarily because I've received very little input from any of you regarding new items.) Please provide me with descriptions of items eligible for inclusion. Devices included in this catalog must have at least one prototype considered to be working properly and must be reproducible from existing documentation. Please send comments, additions and corrections to Ron Cude.

WHOLE ALTO WORLD NEWSLETTER**A RIS BOARD**

This RIS-Board, implements the 9-Wire standard interface for input scanners on an *Alto II*. It's suitable for high speed, raw video input to a Trident or other large data sink, requires its own microcode, and plugs into a standard prewired I/O slot.

INTERFACE: 9-Wire Standard for input scanners
 DOCUMENTATION: [IVY]<Thompson>zRisBd-Rev-#.dm
 AVAILABILITY: 2 built and working. Build your own with a stitch-weld card and above documentation

A GENERAL PURPOSE STICH-WELD BOARD

A universal pattern board for *Alto II* or *Alto I*. It has x columns of x pins plus off grid locations for power and filter caps. It also has provisions for an Augat 122 pin connector on the outboard end. The board is fully supported in the *Sil/Analyze/Route/Build* environment.

INTERFACE: Fits in an Alto card cage
 DOCUMENTATION: Sil documentation and [IVY]<Sil>AltoUnivCard.sil
 AVAILABILITY: Moore Systems, Chatsworth, CA.

A STANDARD INTERFACE

This Standard Interface, provides the bus buffering and microinstruction decoding for a stitchweld board in processor I/O slots. This is derived from the RIS-Board design and is intended as a design aid for amateur interface designers.

INTERFACE: TTL chip level
 DOCUMENTATION: [IVY]<Thompson>StanInt-Rev-A.dm
 AVAILABILITY: Sil-logic diagrams

EIA INTERFACE

This board is a simple serial communications module, *EIA RS232C*, 1200 baud.

INTERFACE: None
 Software: An assembly language driver
 DOCUMENTATION: [ISIS]<NWONG>EIA.memo
 AVAILABILITY: Developed for a special project by ED

If you have any questions please message or call Ngai Wong at 8*823-2258 or 2557.

TOOLS**HARDWARE**

A problem has recently been discovered with *Dysan* 80 megabyte disk packs. These *T-80* packs have had their plastic labels come off while the packs were spinning. This causes the heads to crash, the loss of all of the heads, and the pack. Apparently there was a manufacturing defect in the pack itself. *Dysan* thought they had caught all of the defective packs before they were shipped.

To determine which packs have the faulty labels, look at the top of the pack. If the label is somewhat transparent, it's an "OK" label. If the label is solid white, it may be defective and should be removed immediately.

WHOLE ALTO WORLD NEWSLETTER

If you have experienced this problem, *Dysan* has committed to replacing the packs and paying for the necessary *T-80* repairs. If you have *Dysan* packs that have the white label, Phil Hoffman, 8*923-4624, has some extra labels and has drawn up instructions for their replacement. If anyone needs to replace their "solid white" labels, they can message or call Phil and he'll ship the quantity needed and the instructions for their installation.

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local File Server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [ISIS].

NEW RELEASE: Every software developer in Xerox must have been on vacation this past month. There are no new releases to report in this edition.

RE-RELEASES - SUBSYSTEMS

Would you believe, there haven't even been any updates of subsystems?

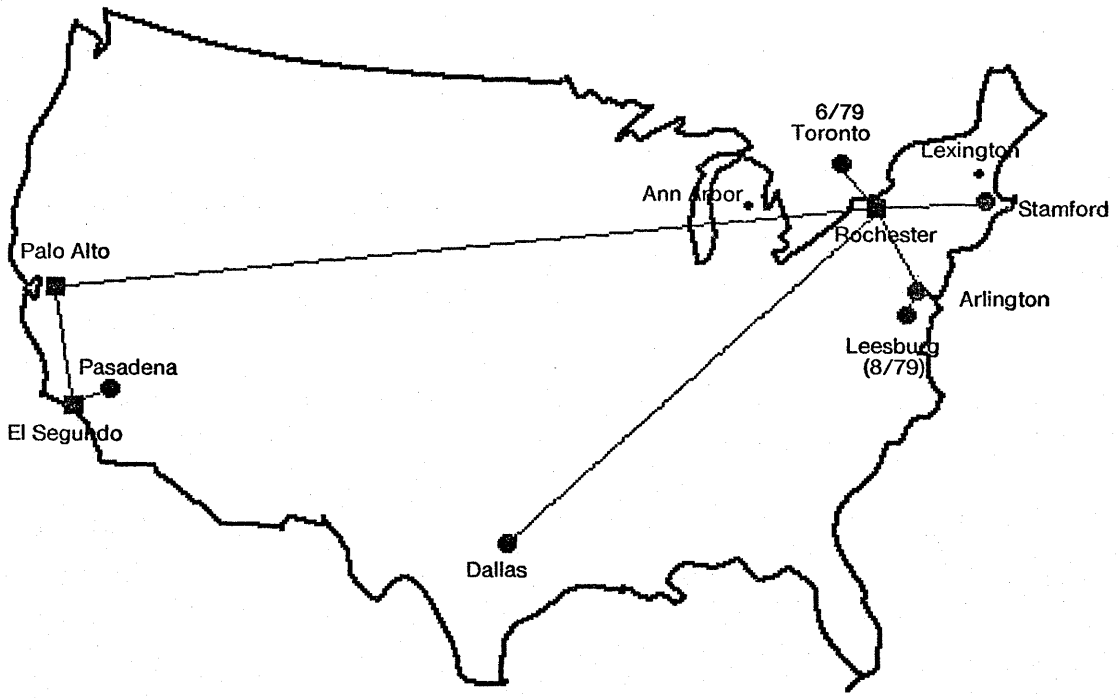
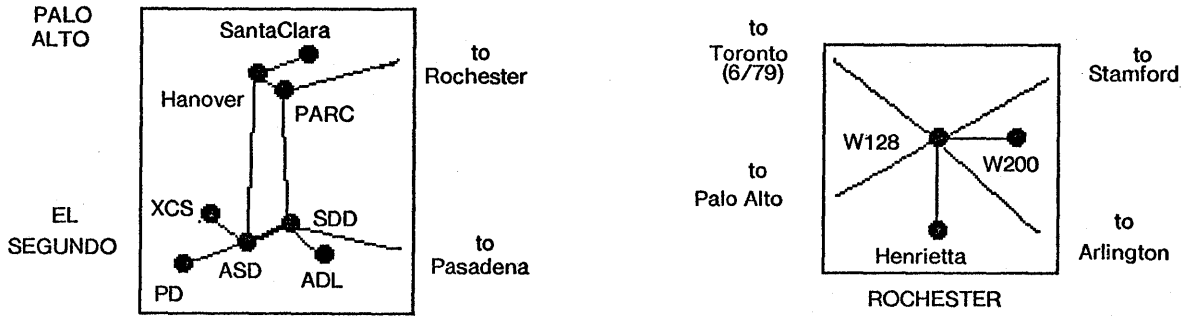
RE-RELEASES - PACKAGES

Since there weren't any updated subsystems, it makes sense that there weren't any updated re-released packages (it makes sense to me, anyway).

TECHNOLOGY

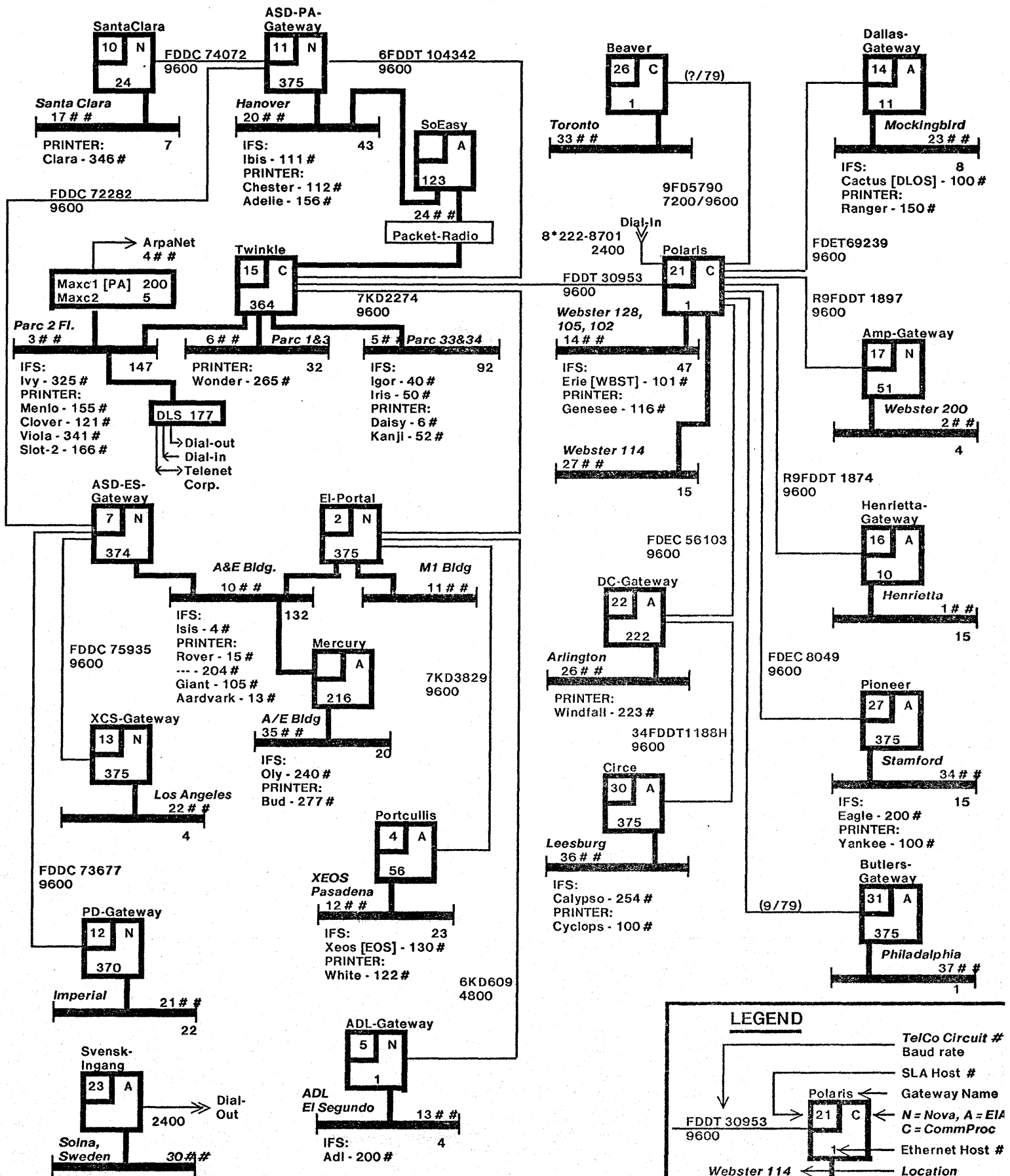
Again, sorry but there are no technology papers this month. Watch this space next month.

The *WHOLE ALTO WORLD NEWSLETTER* is a monthly publication for Xerox employees that use the *Alto*. It is not to be shown to non-Xerox people. Copies are available on [MAXC] and [OLY]<AltoDocs>WawNewsM-YY.Press or may be obtained from the editor, Ron Cude, ASD, by messaging <CUDE> or calling Intelnet 8*823-2465.



NATIONAL ALTO NETWORK
June, 1979

Filed on: [maxc]NetworkMap.press



**ALTO NATIONAL NETWORK
NETWORK TOPOLOGY**
September, 1979

XEROX

Whole ALTO World Newsletter

For Xerox internal use only

CONTENTS

SPECIAL ANNOUNCEMENTS	1
WAW Meeting A Success	1
Welcome to XRCC	1
GENERAL NOTES.....	1
New BootFiles Directory	1
Name Directory Update Policies	1
GSD Announces New Mail Service	2
MARKET PLACE	3
Alto Workstation Multiplexers	3
KeySets For Sale.....	3
Rushmore Offering	3
String Search Routine Needed.....	3
Does Anyone Need A New Clock?.....	3
TOOLS.....	3
Hardware.....	3
Maintenance Notes.....	4
Software.....	4
New Releases	4
Re-Releases - Subsystems.....	5
Re-Releases - Packages.....	8
TECHNOLOGY.....	9
Info '79	10
Rushmore Technical Manual	15

XEROX

Whole ALTO World Newsletter

For Xerox internal use only
Published and edited by Ron Cude
Message <CUDE.PA> or call Internet 8*823-2465

TECHNOLOGY AND TOOLS FOR THE FUTURE

November 16, 1979

SPECIAL ANNOUNCEMENTS

WAW MEETING A SUCCESS

October's *Whole Alto World* meeting went extremely well. I'd like to thank everyone who attended for their contributions and comments. I'd also like to extend thanks to our co-hosts, Darwin Newton and Jim Iverson. The next meeting will be in February, 1980. The exact date and location will be announced as soon as possible.

WELCOME TO XRCC

Joe Wright, Xerox Research Centre of Canada, has announced that their Gateway is up and running. Their message registry is **.XRCC**.

GENERAL NOTES

NEW BOOTFILES DIRECTORY

David Boggs has moved [Ivy]<BootFiles> bodily to [Maxc]<BootFiles> and destroyed the directory on Ivy. <BootFiles> is where almost all of the boot files that one gets from boot servers are kept. Exceptions are DMT.Boot and NewOS.Boot which are kept on [Maxc]<Alto> for historical reasons. The reason for the move is to allow the University Grant people access to them.

Submitted by David Boggs

NAME DIRECTORY UPDATE POLICIES

The following policies will govern maintenance of the Alto Network Name Directory by the Webster Network Support organization:

1. Routine updates will be done each Thursday afternoon (Eastern time). Confirmation will not be sent, unless you request it.
2. Emergency updates will be done within 4 hours, if at all possible; an emergency update may be, for example, one required because a server has failed, and another unit needs to be substituted.

3. All routine requests should be sent to <NetSupport.WBST>. Requests sent elsewhere will have to be re-routed, possibly causing delays.
4. Requests for emergency updates may be phoned in directly. At the present time, for dire emergencies, call Art Axelrod at 8*222-5811. For non-dire emergencies, message <Axelrod.WBST>. In the very near future, other persons will be available. A notice will be sent at that time.
5. As a general rule, requestors should give a second choice name. The directory is now so large that duplications are becoming a problem. Otherwise, in the event of a duplication, a numeral will be appended to your chosen name, e.g "Frodo-II". Alternatively, you are invited to examine PupNetwork.txt to determine whether your pet name is already taken. PupNetwork.Txt currently resides on [Maxc]<System>, but is scheduled to move. A notice will be sent when this occurs.

Don't forget to include your office or lab room number. It is helpful to the maintenance people at your site.

Submitted by Art Axelrod

GSD ANNOUNCES NEW MAIL SERVICE

On November 12, the GSD Mail Center in Monroe County will initiate an experimental Laurel Hardcopy service for non-Laurel Users.

This enables Alto users located virtually anywhere to send messages via Laurel to any Xerox employee in Monroe County by addressing the message to: **Employee Name Mail Stop** <MCMail.WBST>. When using the "cc:" line, copy recipients in Monroe County may be identified by typing: **Employee Name Mail Stop**. Please note, <MCMail.WBST> should not be shown on the "cc" line, otherwise only <MCMail.WBST> will appear on the hardcopy instead of the employee's name and mail stop.

For example:

To: David Lang 209C<MCMail.WBST>
cc: Jim Ferro Conc., Joe Solan W102

Each day, Monroe County Mail personnel will periodically query Laurel for new messages, invoke the requisite "*Hardcopy*" command, and deliver the hardcopy messages. Names and locations on distribution lists containing addresses should be visible. The "Get" command should be used for expanding such .dl's by selecting them and entering "G Esc."

If any message is undeliverable, the Mail Center will notify the sender specified in the "From:" line, via Laurel, with an explanation. Delivery of messages will have a maximum overnight turnaround.

Please direct any comments to Jim Ferro at <MCMail.WBST>, Rochester, Xerox Square Concourse, or INTELNET 8*223-3492.

Submitted by Jim Ferro

MARKET PLACE

Market Place provides a forum for Alto users to make offerings and requests for Alto related hardware and software. To place an "ad" or offering, send the text to the coordinator, Ron Cude, in El Segundo, message <CUDE.PA>, or phone Intelnet 8*823-2465.

ALTO WORKSTATION MULTIPLEXERS

PARC/ADL has a need for at least one Alto Workstation Multiplexer, two if possible. They are prepared to buy 2 new ones but *Cybernex*, the manufacturers, will build only a minimum quantity of 10. Therefore if anyone would like one or more, please let Dave <Cronshaw.PA> know and he will try to get a batch order together. Prompt replies would be appreciated in order to make this years capital budget. Message Dave or call him at Intelnet 8*823-7279.

KEYSETS FOR SALE

Carol Williams, ED/SPG, has announced the availability of 20 Alto Keysets. Anyone who needs a Keyset or two should message <CWilliams.PA> or call Carol on Intelnet 8*823-1654 for more information.

RUSHMORE OFFERING

Rushmore, a module allowing the Alto to be linked to a variety of peripheral interfaces, is announced. A rough draft of the technical manual and ordering information can be found beginning on page 10.

STRING SEARCH ROUTINE NEEDED

Dave <Cronshaw.PA> at PARC/ADL is in need of a string search routine that can find if one string exists within another string. If anyone has, or knows where any such a procedure exists in BCPL, Dave would appreciate hearing about it.

DOES ANYONE NEED A NEW CLOCK?

Bob Lyon of ASD has come up with a new analog clock for the Alto. The run file is at [IBIS]<BLyon>Clock>Clock.Run.

TOOLS

HARDWARE

XEOS ANNOUNCES TWO NEW INTERFACES

XEOS has announced the availability of two new Alto interfaces: an interface between an Alto and a Linotron 202 phototypesetter, and an interface between an Alto and an Optronics drum scanner. Both interfaces make use of the HyType port and hence don't require any modifications to the Alto. Software is available but has not yet been released. Interested parties should contact Chuck Hains at XEOS for details. Message <Hains.EOS> or call Chuck at Intelnet 8*844-2423.

MAINTENANCE NOTES

IC CHANGE ON THE TRIDENT CONTROLLER

It appears that, depending upon the manufacturer, certain 74161 IC's are a bit slower than others. This can cause occasional Trident Controller read and write problems. If you have been experiencing this problem try changing all of the 74161's on the Trident Controller to 74LS161's. This should eliminate the timing problems associated with the slower 74161's. The chips that should be changed are A-13, A-14, A-17, A-27, A-59 and A-61.

Submitted by Phil Hoffman

TRIDENT DISK ROUTINE ERROR

"The Trident disk routines have discovered an unrecoverable disk error." Have you ever wondered what to do when faced by this enigmatic Swat message? Well, try the following approach.

Type xxx + 12 ↑O (while still in swat. xxx is the offending disk command block)

	-- this is the cylinder that caused the problem
LF TAB	-- this is the head, sector of the problem
LF	-- this is the offending drive

Now, wasn't that easy?

Submitted by Bruce Malasky

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local File Server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [ISIS].

NEW RELEASES

★ BRAVOX 3.6 ★

The *caveat emptor* release of BravoX version 3.6 is announced. BravoX is a menu-based, modeless document management system that includes facilities for creating, editing, formatting, filing, transmitting, and printing office documents for correspondence and other purposes. It is similar to Bravo but, being menu-based, is much easier for new users to learn.

To create a BravoX disk with OS 17 and it's related files, retrieve [OLY] or [IBIS] <ASD-Software>XOIS>3.6>OlyMakeXOISDisk.Cm (or IbisMakeXOISDisk.Cm as appropriate) and then do the following:

type @OlyMakeXOISDisk.Cm (return)

This command file will rebuild your disk with a new OS, FTP, and DMT. It will then InstallSwat, retrieve the appropriate BravoX related files and install them. If you are installing BravoX on a disk that has Bravo on it, be sure to delete all Bravo related files before executing the command file.

On [OLY] (or [IBIS]) <ASD-Software>XOIS>3.6>DOCS> you will find several press files. They are:

XOIS-RefManual.Press (551,424 bytes, 143 printed pages)
XOIS-RefManualSupplement.Press (147,968 bytes, 51 printed pages)
TrainingManual.Press (303,104 bytes, 100 printed pages)
Conversion.Press (21,504 bytes, 8 printed pages)

As you can see, the Reference Manual is quite long. It is unlikely that every BravoX user will need their own copy. Perhaps one or two group copies to be used on an as needed basis would be sufficient. The Reference Manual supplements detail the most recent changes to version 3.6 including the new advanced *margins* and *table* menu's.

TrainingManual.Press is a copy of ASD's customer training materials. It will make an excellent reference document and will probably answer most of your questions regarding detailed operation. The practice documents mentioned in the Training Manual can be found on <ASD-Software>XOIS>3.6>TrainingDocs>Document-1, Document-2, Footnote-Exercise, Formats, Fundamentals, Placement-Exercise and Sample-Memo. Please keep in mind that the training document is oriented towards classroom training. However, most experienced Alto user's should find it totally adequate for self-training.

Conversion.Press deals with changes between BravoX and Bravo 8.5. Please read it. It should help clear up a lot of confusion and smooth your conversion process. Most of the differences between BravoX and Bravo 8.5 are also applicable to Bravo 7.4.

And now for the caveat's. This is a development system. Although it is quite robust and is being used extensively, if you choose to use BravoX it is *totally at your own risk*. Old Bravo files can be read into BravoX using the *Command G* option; documents created with BravoX cannot be read into old versions of Bravo. For this reason, during your learning process, don't cast anything in concrete. Please do not bother anyone in ASD with bug reports, questions or any kind of problem. Instead, please send any comments to <ASD-Support.PA>. This account will be queried periodically and the messages dealt with appropriately.

RE-RELEASES - SUBSYSTEMS

★ PREPRESS 1.10 ★

Version 1.10 of Prepress is released as the newer versions of the files:

[Maxc1]<Alto>Prepress.Run
[Maxc1]<AltoSource>PrepressSources.DM.

The old versions will stay around for a while.

There are three new features. First, there is a new window in the top portion of the main menu that displays the type of the current source file, that is, Chars, OrbitChars, Splines, or Widths. There is a new command called DeOrbitize, which converts a bitmap font file from Orbit format (in which the successive scan lines are not aligned on word boundaries) to vanilla *.AC format (in which the scan lines are aligned; AC= AlignedChars, get it?). Finally, there is a new command called ReadAL which will take an Alto font in the *.AL format as input, and produce the corresponding vanilla *.AC file, suitable for use with Prepress. ReadAL tries to guess the family name, point size, and other font data from the name of the *.AL file, which often works. If it doesn't work, you will have to use ReName on the the result. There is also a performance comment: most *.AL files will fit in core, in which case ReadAL will run reasonably rapidly. If your *.AL doesn't fit in core, ReadAL will still work, but it will go much more slowly.

There are also minor changes: this version can use the Trident under OS17, and uses a newer version of the floating point microcode.

The documentation has been brought up to date; a new 31 page manual is stored on

[Maxc1]<AltoDocs>PrePress.Press,

or, if you prefer Bravo format, retrieve the two files

**[Maxc1]<AltoDocs>PrePressA.Bravo
[Maxc1]<AltoDocs>PrePressB.Bravo.**

Submitted by Lyle Ramshaw

★ BRAVO 7.4 ★

With help from Charles Simonyi and Greg Kusnick, and with source files rescued by Bob Lansford, Ed Taft has succeeded in generating a new version of Bravo from the sources. The principal motivation for this release is to adapt to two important environmental changes: the introduction of the new Alto file creation date standard, and the imminent existence of more than 32 networks in the Xerox Internet. (The latter event will cause all earlier versions of Bravo to die horribly during the Hardcopy command.) File servers will be converted to use the new standard in the near future. (See <AltoDocs>FileDates.Press for further information.)

This Bravo should be functionally identical to the previously released Bravo 7.3, but with a few bugs fixed and loaded with the current Pup package (so it won't break when the 33rd network comes up). It is upward-compatible from version 7.3; all earlier versions are hereby obsolete.

Bravo 7.4 also incorporates a few other improvements, most notably the ability to create documents containing color (a feature contributed by Bob Lansford of XEOS). It should be emphasized, however, that Ed Taft is not undertaking regular maintenance of Bravo; please do not report bugs to him unless you can demonstrate that they are newly introduced in Bravo 7.4.

A memo describing the changes in more detail is available in <AltoDocs>Bravo74.Press, which you should send to a color printer if you have access to one. To retrieve the new Bravo itself, you should retrieve the file:

<Alto>Bravo.Cm

and then issue the command:

@Bravo@

(Bravo consists of the files Bravo.Run, Bravo.Error, and Bravo.Messages; it is important that you obtain all three.)

Alto users in Palo Alto may obtain these files from Maxc; if you are outside Palo Alto you should await a local announcement and then obtain Bravo from your local file server.

Submitted by Ed Taft

★ DRAW ★

A new version of Draw.Run has been released on [Maxc]<Alto>. It corrects a problem that caused it not to work under OS 17.

As usual, people outside Palo Alto should await a local announcement and obtain the new version of Draw from their local file servers, not from Maxc.

Submitted by Ed Taft

★ IFS SCAVENGER ★

A new version of IfsScavenger, dated November 2, 1979, is released. Get it from [Maxc]<IFS>IfsScavenger.Run and .Syms. This version cures the "No VMem buffers" bug and a bug in [1-5] having to do with changing the logical unit number when editing Ifs.Home. It requires OS17.

Submitted by David Boggs

★ SIL ★

[IVY] and [MAXC]<SIL>SIL.Run/.Syms contain an update that is purported to correct the stack overflow problem when doing output to a new file. The release date is October 22.

[MAXC1]<SIL> and [IVY]<SIL> clean-up is done. Normally RUN-files, Libraries, Dictionaries, Manuals, Memo's, and DataSheets (press files) are kept in [MAXC1]; the sources are kept in [IVY].

<SIL> has a new program, called *EDBuild.Run*, which is modified from *Build.Run*. The differences are many files created as the result of the *EDBuild* process have revision letters attached with the file names. For example, the wiring list created by *EDBuild* is *Card-C.wl* not *Card01.wl* (if the card is revised from version *B* to version *C*), and the add-delete list is called *Card-BtoC.ad* not *Card01.ad*. See *EDBuild.Memo* for operating instructions. For those who have used *EDBuild* before, please beware that the back-up template file name has been changed from *BuildBackUpTemplate.Cm* to *EDBuildBackUpTemplate.Cm*. This is necessary because *BuildBackUpTemplate.Cm* existed already (it's used by *Build.Run*). As usual, edit *EDBuildBackUpTemplate.Cm* for your own needs.

Submitted by Chuck Thacker & Tom Chang

★ BRAVO 8.5 ★

A revised Bravo 8.5 and Error Processor is released.

The only changes to Bravo 8.5 are to allow it to operate safely under any OS version, 15 or later.

You may find these programs at:

[Ibis]<Bravo>Bravo.Run (Bravo 8.5)
[Ibis]<Bravo>8.5>ErrorProcessor>Error.Run (Error Processor)

After retrieving these files, please execute:

Error/I
Bravo/I

to get them installed properly.

Submitted by Martin Haerberli

★ FTP ★

A new version of Ftp dated 6 October 79 is released. This version automatically retries OPEN commands every five seconds, contains minor bug fixes and improvements but no significant

changes. It is not guaranteed to run under OS16, though it probably will unless you use a Trident. The new version is on [Maxc]<Alto>Ftp.Run. A new version of Ftp.Boot is available from the NetExec. This version is about 25% smaller, which means it is 25% faster to EtherBoot, and it should work correctly with all Alto file system configurations.

Alto users outside Palo Alto: please await an announcement from your local IFS administrator, and then get these subsystems from your local file server.

Submitted by David Boggs

RE-RELEASES - PACKAGES

★ OS 17 ★

Version 17 of the Alto Bcpl Operating System is hereby released. You can update your disk automatically (make sure you have at least 300 free disk pages) by the following steps:

Use FTP to retrieve [Maxc]<Alto>NewOS.Cm.

Type @NewOS

The principal change in this version is that file versions have been removed. Courtesy of Ed Taft, looking up a file is now about 10 times faster. Page 33 of the OS manual [Maxc]<AltoDocs>OS.Press gives details on the other changes.

This material is also available as [Maxc]<AltoDocs>OS17.Tty for those users who don't need the whole OS manual.

Important Note: Mesa programmers should be aware that certain types of Mesa image files have a problem that causes your Alto's clock to stop when you run them under OS 17. This problem may be corrected by re-binding and re-executing Makelimage. Contact SDsupport for further information. Laurel is not affected by this problem.

Submitted by David Boggs

★ SWAT ★

Version 28 of Swat, the BCPL debugger, is released. This version contains minor bug fixes and improvements, but no major changes. Retrieve [Maxc]<Alto>InstallSwat.Run.

Submitted by David Boggs

★ NEPTUNE ★

Courtesy of Keith Knox, Version 3 of Neptune is also released. Neptune is now also available as a boot file from the NetExec. This version will definitely not run under OS16.

Submitted by David Boggs

★ IF ★

The IF subsystem, which stopped working under OS17, has been fixed. The new version is on [Maxc]<Alto>IF.Run.

Submitted by David Boggs

TECHNOLOGY

Featured each month will be articles and papers on technologies affecting or affected by by Altos, new technologies and directions developing within Xerox, and discussions of the work-in-progress within specific organizations.

Included this month is a report by Ginger Engstrom on her recent trip to the 6th. Information Management Exposition and Conference (INFO '79). This interesting paper can be found beginning on page 15.

Also included, as mentioned before, is a rough draft of the Rushmore Technical Manual. See page 10.

The Whole Alto World Newsletter is a monthly publication for Xerox employees that use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC]<AltoDocs>WawNewsM-YY.Press or may be obtained from the editor, Ron Cude, ASD Field Support, by messaging <CUDE.PA> or calling Intelnet 8*823-2465.

XEROX

Business Systems
OIS Marketing & Product Planning

To: **Distribution** Date: **November 7, 1979**

From: **Ginger Engstrom** Org: **OIS/Systems Marketing**
El Segundo/A1-47
Ext: 2098

Subject: **Info '79**

On October 15th thru 18th, I attended the 6th Information Management Exposition and Conference (INFO '79). The show is aimed at the management of information and the techniques and technologies suited to the production, storage, communication and use of information. This DP oriented show is being influenced more and more by the WP and OIS industries.

The only fault I found was that they offered more than one person could attend in the allotted time. The conference was divided into a series of conferences-within-a-conference. The general categories were Information Management, Advances in Information Technology, Office Automation, Small Business Systems, and Industry Oriented Sessions designed for Manufacturing, Banking, Insurance, Hospitals and Legal. The sessions I chose were from the Office Automation area focusing on Office of the Future Planning and Integrated Word and Data Processing.

Most of the speakers at these sessions were users/customers as opposed to vendors or consultants. There was a marked contrast in the level of users at this show from the IWP and other word processing shows. These users were well informed in the problems of office automation from both a technological and sociological point of view. Some of the major companies represented were:

- Exxon Corporation
- Security Pacific National Bank
- RCA Corporation
- Avon Products
- ARCO
- ITT
- Price Waterhouse
- American Broadcasting Companies

All of these companies have an office automation department and the majority of them fall in the same reporting chain as the DP department. A typical organization is an Office Automation Department, a Telecommunications Department and a Data Processing Department all reporting to the Vice President of Finance.

I was impressed by the degree of analysis that has been performed by these various companies and the degree of commonality in their findings. The following sections are a composite of the many companies and speakers represented at the conference.

Most of the companies have been addressing the office automation problem for 3-5 years; many have installed prototype systems and at least one company, Security Pacific Bank, is in the process of designing their own hardware and software for their "Office '85" program. Some of the reasons that more and more companies are aggressively addressing the office automation problem are the costs associated with:

- ★ High turnover
- ★ Government reporting
- ★ Communication cost (especially USMail)
- ★ Increasing litigation
- ★ Increasing complexity in large, diverse companies

They have found that 80% of all costs in an office are associated with supervisory and management positions, while only 20% with clerical. Savings in an office must come by making the supervisory and management segments more productive. This will be done by providing timely, quality information in formats flexible enough to respond to the complex and rapidly changing business environments.

What do they want in a system? Probably the major concept wanted in a automated system is INTEGRATION, the ability to pass data from one function/application/system to another without consciously thinking about it. I believe this to be a major change in user thinking that will make the Star architecture invaluable in the industry; **"The world is really ready for us"**.

A system must also be interruptable; the ability to start multiple functions without terminating previous ones and returning to them when desired.

All the companies represented felt the need to be able to build future OIS systems on existing office systems. It is not cost effective for a company to replace an entire Telex network, an IBM or DEC mainframe, a 9700 or a Mergenthaler printer with vendor specific solutions to their problems. Corporate ITT has selected Wang as their vendor because of its ability to interface with many external protocols and devices but more importantly because of its programming ability to write interfaces when none exist. The following applications are specific to ITT, but are typical of the other companies.

- ★ Corporate speech writers inputting text, sharing files and doing research using remote timeshare databases.
- ★ Administrative secretaries inputting parameters for organization charts via a high level user interface program on a Wang WP System for input to a 9700 via the IBM mainframe.
- ★ Creation of the text portion of the annual report on a Wang WP system for final input to a Compugraphics typesetter.
- ★ An electronic mail system 'HOBBIT' running on the IBM mainframe outputting to Wang, Telex, Datapoint terminals, other WP systems and other networks.
- ★ Customer service clerk inquiring into an existing microfiche database via a Wang WP System.

Following is a composite list of tasks that most users feel are required in an integrated office environment:

- Text Editing - Initial inputting
- Text Processing - Revision & formatting
- Electronic Mail - Interface with other networks and other terminals.
- Electronic Filing - Cross Indexing and multiple keywords
- Records processing
- Data Entry
- Communication - Pass thru capability (3270)
- Programmer Development Tool (offload TSO)

Calendar Management
Correspondence Tracking
Tickler/followup
To do lists
Phone Messages
Diary
Bulletin Board
Decision Making Aids

Graphics/color
Help
Security
Integration
Interruptability

It is interesting to note that these large users are not expecting an OIS system to replace their existing DP equipment. Interfaces are required but they recognize a need for both areas. DP is oriented to running structured volume jobs. OIS systems are expected to supply the flexibility required in current office environments. At ITT, their telephone directory application is maintained on the Wang WP System, sent to the IBM main frame for sorting and then to the 9700 for formatting and printing. When the entire application was performed on the Wang system using an impact printer, time to complete was 8-10 hours. The current application takes less than 1 hour.

What has been implemented? Most of the companies have installed probes and/or pilot departments. The most popular vendor is Wang because of their broad sales coverage (national & international) and their wide range of products and interfaces. Electronic mail systems are running in most of the companies using packages like Hobbit and ComNet on main frame computers. As I mentioned earlier a successful mail system must interface with existing terminals and mail systems already installed. Many companies have corporate databases of correspondence, reference material and historical data. These also have been implemented on their central mainframe with cross indexing and elaborate keyword structures. The Wang systems have been programmed to access these systems. Office aids, including calendar, scheduling, etc. are currently being integrated using in-house programmers.

It was the general feeling among the users that it would be five years before they were effectively utilizing the hardware and software available today. The basic problems in implementing an office of the future are:

1. Lack of qualified people. Qualified people are defined as trained systems people without a DP orientation. One user explained this qualifier in the following manner. "DP people are trained to deal with structured input and output and use hardware efficiently, OIS people must deal with unstructured input and output and use people efficiently".
2. Lack of vendor support. The general attitude towards vendors is; plan to do everything yourself and believe nothing they tell you.

While most users are buying existing systems and integrating them into their environment, Security Pacific is taking a different approach. They have talked to all the leading vendors and realize that the equipment being designed today is oriented around a traditional keyboard and text-editing systems. For true acceptance by the professional environment they feel that it is necessary to design equipment specifically for the management of information.

In conjunction with California Design Institute, they are designing a terminal with the following specifications:

Cost of \$2,000
 Compact - desktop
 Speaker Telephone with auto dial of selected numbers
 Electronic Calculator
 Date & Time Display
 Voice Message Recorder
 Function Keyboard
 Alpha Entry Capability

Software available on the system will incorporate electronic mail, electronic filing, document retrieval, calendar, tickler, telephone aids and many other office functions.

EQUIPMENT

It was interesting to note that the vendors who exhibited at Info overlap tremendously with those exhibiting at WP shows; text editors, dictation and fax have joined the larger computers, timesharing systems and software houses.

As of today, no one is providing all the features/tasks that the leading users feel they need. Star comes closer than anyone to fulfilling those needs especially in the areas of integration and interruptability. Our weak areas appear to be interfaces to foreign equipment and user programming.

New and/or interesting products at the show included:

IBM Electronic Typewriter 75

IBM announced a new super typewriter that falls between their 50/60 and their memory typewriters. Based on a high density 36,000 bit random access memory chip, the unit can provide 7,500 or 15,500 characters of storage. The cost is \$2075/110 for the basic unit and \$2300/\$120 for the expanded model. It provides document and phrase retrieval, revision capabilities and formatting.

QYX Level One Plus

QYX, not to be outdone by the IBM announcement, announced their Level One Plus providing 8,000 characters of storage for \$2,100. It distinguishes itself from the IBM offering through its optional display and communications capability.

CPT

CPT, one of the more popular stand alone text editors, recently announced a shared resource system consisting of up to 8 stand alone systems connected to a central processing systems with 25MB or 50MB Winchester disks. Besides offering expanded file capability it provides additional 'background' capabilities for sorting, merging, printer queue management and other tasks requiring minimum operator interaction. With this announcement, CPT joins Lanier in following the successful Wang building block sales strategy; "Start anywhere within our product line and enhance the hardware and software as your needs grow." CPT also announced their CompuPak capability which will allow qualified software houses to supply support, application packages and compilers to their customers. This provides Basic, Cobol and Fortran interfaces for their systems.

Lanier

Lanier has also moved into the shared-resource market, allowing their No Problem systems, with or without disk drives, to be attached to a CPU capable of supporting

additional disk storage, printers, OCR, photocomposition and electronic mail. Intelligent and dumb terminals may be mixed on a system. Peripherals may exist at a user station or be attached at the CPU. According to the press announcement, a user now has more than 1,000 systems combinations to choose from to meet the needs of small, medium and large sized firms. Again a user can buy small and grow big without replacing hardware.

Artec Dual Display

Artec was displaying a prototype of their new dual display. The full page screen is attached to the Artec single line display system. The single line can be used as a prompting area, background processing window or as an aid for data entry. The full page screen is extremely clear and stable showing proportionally spaced justified copy as well as bold and super/subscripts. The two displays, keyboard and processor cost 11K. The components can be mixed and matched in a number of ways; the display keyboard can function as a complete input and editing system or it can be combined with the full page module for more powerful formatting and editing functions. Up to 8 stations can be configured to share a central file manager.

RUSHMORE TECHNICAL MANUAL - (Rough Draft)

October 1, 1979

Introduction to the Alto Interface

This card links an Alto computer to a variety of peripheral interfaces. The design basically consists of a Z-80 microprocessor with up to 16K bytes of dynamic RAM, several special purpose peripheral interfaces accessible as memory locations by the Z-80, and a facility to allow the Alto to directly read and write any location in the Z-80 memory address space. Thus, the Alto has direct access to all the control and data words for each peripheral, and may also communicate with the Z-80 by direct access to locations in the RAM. In the later case, the Alto can load the RAM with programs and data which will make the Z-80 work as channel controller. From the viewpoint of the Alto programmer the interface consists of several dedicated locations in the Alto memory space, through which are transferred data consisting of command words, status words, addresses in the Z-80 memory space, and byte pairs to be read or written at that Z-80 address.

The peripheral interfaces are:

- ★ an RS-232C interface with both primary and secondary channels, hardware assist for the receive data, and distinct programmable baud rates for transmit and receive;
- ★ a CBS-compatible phone interface using a filtered DAC to generate tones, and can also be used as a general purpose programmable analog output;
- ★ a complete IEEE-488 bus interface, with hardware assist for the talker/listener/controller enables, and this can be used as a general purpose 8-bit port suitable for driving a Summagraphics BITPAD device;
- ★ an external Z-80 bus which will support DMA controllers, and will also support those S100 compatible devices which are not bus controllers, and which may share the same physical wires for input and output data.

In addition to the above, there is a baud rate generator which can be enabled to interrupt the Z-80 at 9600 HZ, and various bus arbitration logic which allows a controller on the external bus to directly access any Z-80 memory location in the system. Also, the Alto may directly perform various tasks such as reset of the interface card, testing and setting dedicated flags, and enabling a task WAKEUP signal which can be asserted under program control by the Z-80.

The interface card has been designed to work in one of the Alto memory interface slots (5 and 6) of the Alto mainframe, and requires only four additional wires on the backplane. Connectors on the exterior end of the card support separate cables for each of the four peripheral interfaces. A jumper on the card may be used to change one bit of its Alto address, so that two such interface cards may be used simultaneously.

The main Z-80 memory is dynamic RAM, which is automatically refreshed by the Z-80 when it has control of the bus. For RAMS which have a maximum refresh time of r ms., there should be an average of no more than $22.5 * r$ Z-80 T-states, or $45 * r$ Alto microcode cycles, between the starts of Z-80 instruction fetches.

Interface Card -- Common Control

The address space of the interface card can be accessed from any one of three sources: the Z-80, the Alto, or a Z-80-type controller on the external bus (X-bus). Only one of these sources has control of the (data buses), (address buses), (memory control signals) at one time. Logic drawing sheet (LDS) #11 contains the arbitration and timing logic for handling bus requests from the Alto and from the X-bus: if the X-bus does not already have control, and the Alto requests control, then the Z-80 releases the bus to the Alto. If the Alto requests control while the X-bus is in control, then control will transfer directly to the Alto when the X-bus is done, without the Z-80 regaining control. When the Alto is not requesting or using the bus, then the X-bus request can force the Z-80 to yield control at any time in the usual manner. Note that refresh of the interface card dynamic RAM occurs only while the Z-80 has control. This feature, along with the specified maximum wait for an Alto access, implies that any X-bus DMA device steal only single memory cycles. The bus-acknowledge signals are appropriately clocked so that they can be used to enable data and addresses onto the busses during the necessary decode, setup, and hold times of the data transfer. The X-bus acknowledge is clocked so as to prevent a glitch occurring when the Alto has just finished using the bus, the Z-80 has not yet established control, and the X-bus has just sent a bus request. In this case the Z-80 will perform one machine cycle before going control to the X-bus.

Alto Access

The Alto can read or write any two-byte location in the interface card address space. To facilitate this, there is a 15-bit address register and a 16-bit data register, and a read/write flag which can be loaded directly from the Alto. There is also a 16-bit data register which can be directly read by the Alto.

To perform a read operation, the Alto directly loads the 15-bit address register, while simultaneously setting the read flag. Loading the address register sets the Alto bus request flag. When the Alto obtains control of the bus, a memory sequencer circuit begins generation of signals which emulate two memory read cycles of a Z-80; one cycle for each of the two bytes which are loaded into the Alto data buffer register. When the second memory read cycle finishes, the Alto bus request is reset, and the Alto may directly read the data buffer register at any later time. A write operation is similar. The Alto loads the data register with one 16-bit word, then loads the address buffer while setting the write flag. The Alto bus acknowledge is again set, then reset when the second byte of the data word has been written out to memory. The data register for input is physically distinct from the output register; however, they are each accessed at the same Alto address. The logic which emulates the Z-80 memory timing signals generates a write strobe, AWR, which is gated to the common interface control bus only during an Alto write operation. If the Alto logic is doing a two-byte read operation, this strobe is used to clock the data into the buffer registers for latter access by the Alto. During both read and write operations, the address is switched to that of the odd byte at the trailing edge of the first AWR strobe pulse. However, during a write operation the even byte is kept on the data bus long enough after the trailing edge of AWR to satisfy hold and skew times at the device into which the data was written. This data transfer timing logic tests the common WAIT signal in a manner identical to the Z-80.

The Alto may also directly write a command word to the interface card, and may directly read a set of 8 status bits.

Main Memory 4K (16K) Dynamic RAM

The main memory does a RAS cycle whenever there is a memory operation in progress on the main bus, even if the address is not specifically the internal RAM. The only exception to this is when RESET is enabled. Since giving a short RAS pulse may damage the contents of the RAM, it is advisable for the DMA controller on the X-bus to always provide an MREQ signal of not less than the minimum RAS width of the memory chips being used. (For this same reason,

when RESET is requested by the Alto command bit, it is held back from becoming active while a memory cycle is starting.)

A memory cycle always begins with the low order address bits (row address) gated to the RAM chips through a multiplexor. If the total address is to the RAM, and it is not a refresh cycle, the upper address bits (column address) will be switched onto the chip address lines on the clock after RAS begins. An R-C time delay is provided to make sure the CAS pulse is correctly delayed from the row/column address switch.

At the row/column multiplexor, a high order address bit is jumper selected so that when using 4K RAMs instead of 16Ks, one of the row address bits may be used as a column address bit.

The memory data inputs are asserted low and are not suited to directly driving the bus, so the outputs are routed back to the common data bus through an inverting multiplexor. The other input of the multiplexor is a constant Z-80 HALT instruction. The select input is the halt flag which is loaded by an Alto command write, so the Alto can force the Z-80 to receive a HALT instruction during a normal instruction fetch.

External Bus X-Bus

Bidirectional bus transceivers are used to interface the internal and external data and address lines. The address transceivers are always enabled, with their direction dependent only on whether the X-bus is in control. The data transceiver is enabled to drive a bus only when an external X-bus controller must access a location on the interface card, or when either the Z-80 or the Alto wants to access a memory location on the X-bus, or when the Z-80 is doing an I/O operation. The direction control for data depends on whether the operation is input or output, and from which side the request was generated. The Alto can access the X-bus memory address space exactly as the Z-80 does, but may not do I/O (not-memory-addressed) accesses on the bus. An X-bus controller may access any memory location internal to the interface card, but may not drive any I/O port access controls on the card. (All locations internal to the interface card are addressed as memory locations, not I/O ports.)

The system will support any Z-80 compatible controller on the X-bus, but S-100 compatible (8080 type) controllers can be connected only if they do not require DMA capability, and they can work with physically connected input and output data lines. Some of the X-bus control signals are always enabled, while others are enabled only for S-100 signal mode or for Z-80 signal mode. Since there are four Z-80-only signals which use the same physical bus lines as four S-100-only signals, the design does not, in general, support using both Z-80 devices and S-100 devices on the X-bus at the same time. The Alto can select which set of signals are enabled by setting a flag with a direct command write. No internal vectored interrupt is implemented, but if an interrupt address is provided by the Z-80 or S-100 peripherals, a vectored interrupt mode may be used.

The interface board memory data transfer is controlled by three lines MREQ, RD, and WR. Either the Z-80 or the Alto memory timing logic can drive these internal lines directly when in control of the bus, and they are enabled onto the X-bus. These lines are also address inputs to a 32X8 PROM which derives the corresponding S-100 signals, which can then be enabled onto the X-bus. When a Z-80-type device on the X-bus is in control, it drives the external Z-80 control signals, which are enabled onto the corresponding internal control lines, permitting an external Z-80 type controller to access all the memory locations on the X-bus, as well as those on the interface board.

The S-100 signals $\emptyset/1$ and $\emptyset/2$ are generated so as to minimize skew with the internal Z-80 clock, as shown an LDS #1. $\emptyset/2$ is opposite in phase to the Z-80 clock, and an RC delay is used to switch $\emptyset/1$ high about one half Alto clock period before $\emptyset/2$. $\emptyset/2$ is gated by $\emptyset/1$, so they will not overlap unless due to excessive signal skew on the bus. With a Z-80 cycle time of 340 ns., and with typical tolerances in the electrical characteristics of the time delay circuit, the $\emptyset/1$ and $\emptyset/2$ signals are suitable for driving an 8080A-1, but may not correctly drive the slower 8080 types.

The interface card generates a PSYNC pulse at the beginning of each memory or I/O port signal, and enables it to the S-100 devices on the X-bus with the basic control signals. Since the interface care Z-80 or Alto logic causes PSYNC to be started later in their cycle than would be generated by an 8080, the PSYNC pulse is used to make an artificial "wait" signal for one cycle, so as to give the external S-100 device time to respond to the access request. This automatic wait can be enabled/disabled by a flag set by an Alto cammand write. Both the PSYNC wait and the normal WAIT are disenabled when the address is not external.

EIA RS232-C : PHONE INTERFACE

The EIA drivers will use +/-9.0 volts, rather than +/-12.0 volts, in order to keep the chips' power dissipation low. Two of the signals, Off-Hook and Data Terminal Ready, are inverted in sense at the data register so that when the power-on-reset clears the register, the signals are forced to their OFF state. For the transmitted EIA data, one bit is clocked into a flip-flop, which is then synchronized to 9600 Hz to reduce jitter. One bit in the command word is the EIA Transmit Clock enable. In asynchronous mode this bit can be always turned off. In synchrous mode, we assume that the Alto has set a flag with a command write which enables the Z-80 to be interrupted on the positive edge of the 9600 Hz signal. Depending on the speed of transmission, the Z-80 can either enable the Transmit Clock for one negative pulse in the middle of the data period, or (for 9600 Baud) he can leave the clock always enabled, so the ON to OFF transition occurs between data value transition.

The 7524 is an 8-bit DAC, whose output drives a filter amplifier for signal conditioning. Two input lines are shared between the RS232-C and the phone signals. We are assuming that only a CBS type DAA will be used. Two of the inputs, SH and line current sense, are read through the memory location reserved for the Alto status word.

ORDERING INFORMATION

From: Ted Stollo

Deliveries of Rushmore Cards should be approx 1Q80. The items are expensed according to the definitions of Corp Research Control although their cost is >\$500 per unit. You may want to check with your local controller for a reading on this. However, for people in corporate research, it seems clear that orders will come out of 1980 operating/expense funds.

The Delivery queue is intended to be first in/first out. It is therefore worth getting your orders in soon.

If you are ordering from PARC, simply fill out a blanket order release with Systems Concepts Inc. as the vendor using the attached PO for price info. If you are from Xerox but not PARC, observe the following instructions.

From: Mike Levitt (SYSTEMS CONCEPTS INC)
Date: October 1, 1979
Subject: Placing orders for RUSHMORE cards

This is a list of all the information which Systems Concepts Inc will need to properly process an order for RUSHMORE cards. A brief explanation has been included for each item. Please reference the Xerox PARC PO number for consolidation purposes and alignment with terms/conditions/prices, and quantity discounts.

1. Date of order
2. Purchase order number (to be used as a reference for invoicing, etc.)
3. Quantity (number of cards being ordered)
4. Release schedule (desired delivery date for each card ordered)

5. Purchasing contact (name of individual handling contractual details)
6. Purchasing address (mailing address and phone number of contact)
7. Shipping contact (name of individual who is to receive card)
8. Shipping address (complete address including any special mail stop, etc.)
9. Shipping instructions (will ship all items UPS unless otherwise specified)
10. Billing contact (name of individual who will process invoice)
11. Billing address (complete address including any special mail stop, etc.)
12. Special instructions (anything not covered by nos. 1 through 11 above)

Please be aware that the minimum release is 10 (ten) units, delivery is FOB origin, and our terms are net 30 days.

Please address purchase orders to

David Renton
Systems Concepts Inc
520 Third Street
San Francisco, Ca 94107
415-442-1500 or 415-941-2221

XEROX

Whole ALTO World Newsletter

For Xerox internal use only
Published and edited by Don Winter
Message <DonWinter.EOS> or call Intelnet 8*844-1064

TECHNOLOGY AND TOOLS FOR THE FUTURE

January 31, 1980

SPECIAL ANNOUNCEMENTS

The Next Meeting

Current plans call for the next Whole Alto World meeting to take place in the Palo Alto area in late February or early to mid-March. The details are still being worked out. A specific announcement will be made when all the details are available.

GENERAL NOTES

Alto Fonts

A complete series of Alto TimesRoman and Helvetica Fonts in Extended Character sets has been finished. They are on [XEOS]<Fonts>TimesRomanE*.al and HelveticaE*.al and also on [XEOS] <Fontcenter>Alto>TimesRomanE*.al and HelveticaE*.al. The point sizes are 8, 9, 10, 11, 12, 13, 14, 16, and 18.

I have decided to use the "E" in TimesRomanE to mean Extended character sets (with characters above Octal 200) rather than extended widths.

Please let me know your comments and suggestions on these fonts.

Submitted by Ron Pellar

Additional Uses of a Press "Server"

With respect to the new 2600 printer I note with interest that the proposed system will be driven by PRESS rather than by SPRUCE. If this is to be the general case people considering getting such a device may like to hear of the way we here at ADL have made our TC200 printer server function since it also is PRESS driven.

As was mentioned in the item concerning the 2600, EFTP is the suggested utility for handling the file transfers. That is the way we started off but quickly discovered that EFTP is somewhat limited and I wrote a new version of it called AFTP (ADL's EFTP) which while remaining compatible with EFTP provided for the transfer of the file leader pages to preserve names etc. The result of this is that the file name extension can be used to determine the set of operations which are to be invoked on the server, printing being just one of them. Since it spends only about 10% of its time printing on the TC200 this results in good usage of the machine most of the time.

Whole ALTO World Newsletter

Things it can do for us are:

1. Cause a .press file to be printed on the TC200 by PRESS
2. Cause a .bravo file to be proofread by PROOFREADER, the results being sent back to the sender of the document.
3. Cause .bcpl, .txt and other similar files to be printed on an attached old fashioned line printer over the EIA interface board (300 lpm)
4. Files sent by programs such as Bravo and Laurel using the HARDCOPY command do not contain leader pages. They are checked for having a valid PRESS password and then printed on the TC200.
5. Executes the commands stored within any Line.cm file received. This allows the server to act as a slave for the user and since no controls are imposed on the contents of such files it is only suitable for use in a small "friendly" group.

All of this is possible through the voluntary standardisation of the file name extensions of the files sent to the server. AFTP is the receiving program, after the files are received IF is run to test the file name extensions of the received files and the appropriate set of commands is then executed. The PAGING.PG file for proofreader and other large files are kept on the TRIDENT and are pulled down with TFU as they are needed. A typical printing job or proofreading job (loved by the secretaries) takes less than 5 minutes, mostly in PRESS or PROOFREADER.

People who are interested in setting up such a server should message me for further information.

Submitted by Dave Cronshaw

MARKET PLACE

Market Place provides a forum for Alto users to make offerings and requests for Alto related hardware and software. To place an "ad" or offering, send the text to the editor, Don Winter, in Pasadena, message <DonWinter.EOS>, or phone Intelnet 8*844-1064.

TOOLS

MAINTENANCE NOTES

Can't run XMesa5.0 in newer altos

We recently discovered a problem that should be passed along.

Symptom: After installing XMesa5.0 microcode and inserting the proper switch (SW2) into U-51, XMesa programs will not run.

Cause: Chip U-50 on the 2K Control Module has a wrong chip installed. It should be a 74S153. We found 4 out of 5 recently shipped altos to have a 74S157 installed instead.

We have inserted a 74153 and it has worked so far. This is recommended if an "S" type is not available but should probably be changed as soon as an "S" type can be obtained.

Because of the high number received it is likely that this may be the cause if you have XMesa problems. Good hunting!

Submitted by Phil Hoffmann

Whole ALTO World Newsletter

SOFTWARE

In general, the subsystems, packages, and documentation indicated here will be available from your local File Server under the directories <Alto> and <AltoDocs>. If they are not available, or if you are in doubt as to the version, they may be retrieved from [MAXC] (same directories). Files stored under other directories are on [MAXC] unless otherwise indicated, e.g. [ISIS].

NEW RELEASES

MenuFirst Retrieval System

The New MenuFirst document retrieval system is now being released. MenuFirst allows users to do free text searching on document abstracts, memos, mail files, etc. In addition, several public data files are already available for searching (including the Xerox Disclosure Journal and Communications of the ACM).

What's new about this version (3.2) of MenuFirst?

1. It's called XMMenuFirst and requires at least a 128K Alto to run.
2. Extended memory is used for the display bitmap, thereby increasing the amount of display space for documents.
3. Routines are now provided for users to create their own databases; either on the local disk or on a special Wifs server (CASK) running in Webster. In particular, Laurel mail files can be used as input to MenuFirst.
4. It is incompatible with any previously released versions of MenuFirst (except version 3.1).

A 3 page document on MenuFirst is available on
[ERIE]<Wifs>MenuFirst.doc.press

Please report all bugs, comments, suggestions, questions, etc. to Dattola.WBST and Sauvain.WBST.

Submitted by Bob Dattola

Mesa Floating-Point Package

An Alto-Mesa 5.0 extended-precision floating-point arithmetic package has been developed. It's somewhat slow, but guaranteed accurate according to Knuth's algorithms. It's currently compiled to operate with a 48-bit fraction (mantissa), and could also be re-compiled to operate with any 16-bit multiple not less than 32. (If less than 32-bits precision is required, e.g. to conform to Mesa's 2-word REAL number data type, some other software should be used.) The relevant files are dFloatDefs and dFloat, in [ISIS]<UserInterface>dFloat.dm.

Submitted by Paul Gloger

RE-RELEASES - SUBSYSTEMS

Neptune 3.1

A maintenance release of Neptune.run and Neptune.boot incorporating the new Trident software is available. This is version 3.1, dated 25 November 1979. Relevant software is on:

[Maxc]<Alto>Neptune.run
[Maxc]<Alto>Neptune.syms
[Maxc]<AltoSource>NeptuneSources.dm

Whole ALTO World Newsletter

This new software only affects those using the Trident features.

Submitted by Keith Knox

Mesa 5.0 patches

We have made several patches to Mesa 5.0 software required by the recent Teak and Tools 5.1 releases; the following subsystems are now available on [Iris]<Mesa>Temp> (and will appear on Ibis, Isis, and Ivy shortly):

```
BasicMesa.*
Binder.*
Doc>Mesa51.press
Fetch.bcd
Mesa.*
System>ImageMaker.*
Utilities>IncludeChecker.bcd
XDebug.*
XMesa>BasicXMesa.*
XMesa>XDebug>XCoreMap.bcd
XMesa>XDebug>XMDebug.bcd
XMesa>XMesa.*
```

The changes are described in Mesa51.press and are generally limited to localized well understood bug fixes (changes to the IncludeChecker are more extensive). Note, however, that this software has not gone through the normal six week alpha testing period; this is the reason for the Temp> subdirectory, so that the original release is still available. Use of the new software is entirely optional.

If you encounter problems, send a change request to SDSupport in the usual way; please identify this software as version 5.1.

Submitted by John Wick

RE-RELEASES - PACKAGES

Pup and Ftp 6.0a

New versions of the Mesa Pup and Ftp packages have been developed in conjunction with the University Grant Program; the following packages are now available on [Igor]<AlphaMesa> (and will appear on Ibis and Isis shortly):

```
Doc>FTPPackage.press
Doc>PupFtp60a.press
Doc>PupPackage.press
FatPup>*
FTP>*
Pup>*
```

The major change has been to remove product-related features from these packages; there are some minor changes in the interfaces and several bugs have been fixed. These changes are described in PupFtp60a.press.

Whole ALTO World Newsletter

All of these packages have been recompiled (including the interfaces). However, conversion to this version is entirely optional; it will remain on <AlphaMesa> until Mesa 6.0 is released, to avoid disturbing the 5.0 release on <Mesa>.

If you encounter problems, send a change request to SDSupport in the usual way; please identify this software as version 6.0a.

Submitted by John Wick

The Whole Alto World Newsletter is a monthly publication for Xerox employees who use the Alto. It is not to be shown to non-Xerox people. Copies are available on [MAXC]<AltoDocs>WawNewsM-YY.Press or may be obtained from the editor, Don Winter, XEOS, by messaging <DonWinter.EOS> or calling Intelnet 8*844-1064.
