Parallel Programs from Constraint Specifications

Ajita John

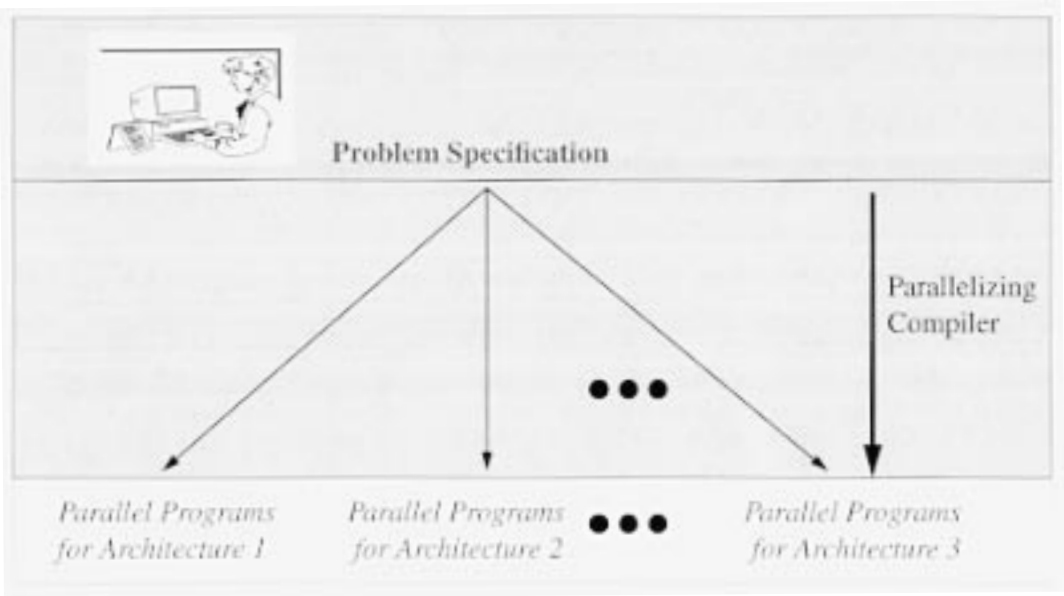
Department of Computer Sciences

University of Texas at Austin

Advisor: Prof. J.C. Browne

ajohn@cs.utexas.edu
http://www.cs.utexas.edu/users/ajohn

Application Oriented High-level Specification of Parallel Programs



Goal

High-level Specifications ---> High Performance

A representation for problem specification which

- places minimum burden on user
  - no explicit parallelism in specification
- allows extraction of maximal parallelism
- allows reuse of components

Constraints:

attractive choice for a representation for problem specification

Why ?

Goal

Extraction of efficient parallel structures from constraints

Contributions

1. Programming system based on constraints for matrix computation.
    - *Hierarchical Type System*
    - *Programming Representation*
    - *Modular Structures for Constraint Systems*
    - *Basic Compilation Algorithm*
    - *Extension for Cyclic Constraints*
    - *Extraction of Task and Data Parallelism*
2. Translation to multiple parallel architectures.
3. Development of applications yielding high performance.

A constraint is a relationship between a set of variables.

E.g. 1

```
C == (F-32) * 5/9
```

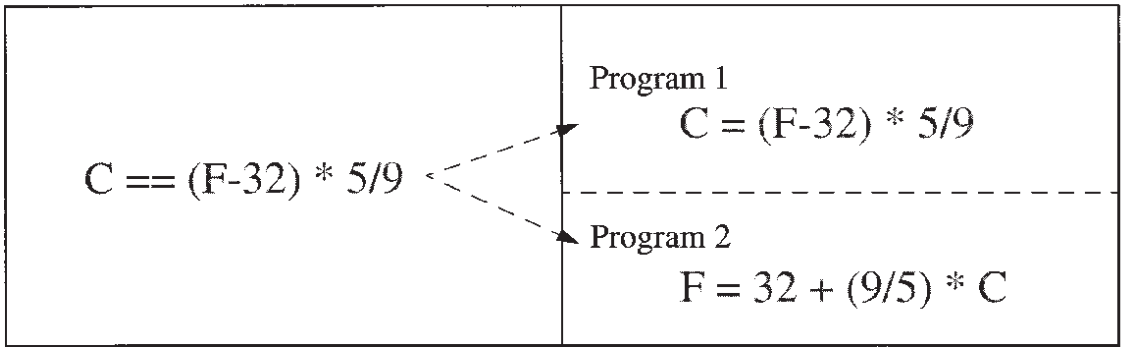
E.g. 2

```
F == (9/5) * C + 32
```

== is equality as opposed to assignment

1 ≡ 2

Constraint Programming vs. Imperative Programming



Related Work (Representative Examples)

Constraint Programming

- *Thinglab - Alan Borning*
- *Consul - Doug Baldwin*
- *CCP, Oz - Vijay Saraswat, Gert Smolka et. al.*

Parallel Programming

Imperative Programming

- *HPF - Ken Kennedy et. al.*

Equational Programming

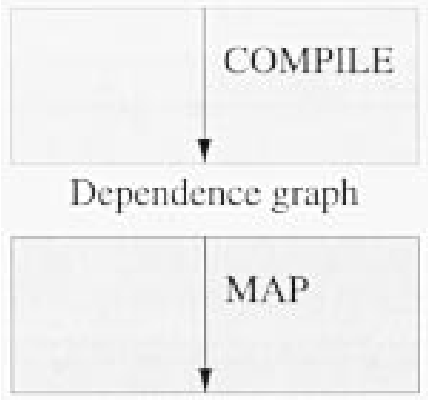
- *Unity - Chandy, Misra*

Logic Programming

- *PCN - Chandy, Taylor*
- *Strand - Foster, Taylor*

Approach

Constraint program (constraint specification & input set)

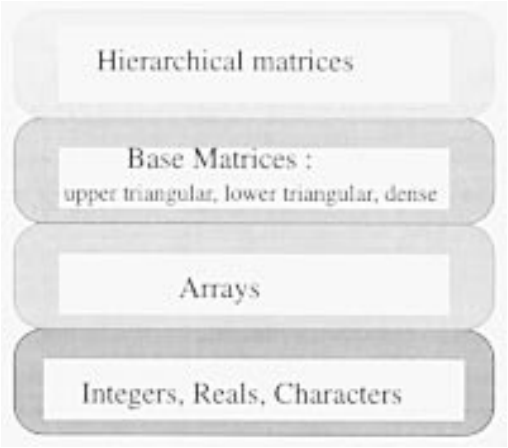


Sequential and parallel C programs generated for

CRAY J90, SPARCcenter 2000, Sequent,
PVM,
MPI (under development)

CODE - Browne, Newton et. al.

Hierarchical Type System



- Constructs new types at higher levels from existing types at lower levels.
- Provides a tool for controlling granularity.
- Each high level type may incorporate domain information in its definition.
- Different efficient algorithms can implement operations depending on the type.
- We have a translator for specifications over a rich matrix type set to parallel programs.

Operator Expressions

- the building blocks for constraints

Expressions involving:

1. Scalar operators, pure function invocations

   E.g. $a + b * sqr(c)$

2. Matrix operators

   E.g. $M_1 * M_2 - M_3$

3. Indexed operators: <op> FOR (<index> <b1> <b2>) <expression>

   E.g. $+ FOR\ (i\ \ 1\ \ 5)\ \ A[i]$

   Equivalent to $A[1] + A[2] + A[3] + A[4] + A[5]$

Constraint Representation

Simple Constraints: relational operators (<, <=, >, >=, ==, !=) over operator expressions
E.g. $a + b > c$
$M_1 * M_2 == M_3$ | only == for matrices

Propositional Operators: NOT, AND, OR, on constraints:
E.g. $a + b > c$ *AND* $a == 5$

Constraints over indexed sets:
AND/OR FOR (<index> <b1> <b2>) { $C_1, C_2, \ldots, C_n$ } C_i is a constraint
E.g. *AND FOR (i 1 5) { A[i] == A[i-1]+1, B[i] == A[i] }*
- represents 10 constraints

Constraint Modules: encapsulation of constraints over parameters
E.g. *EquivalentTemp(F,C)*

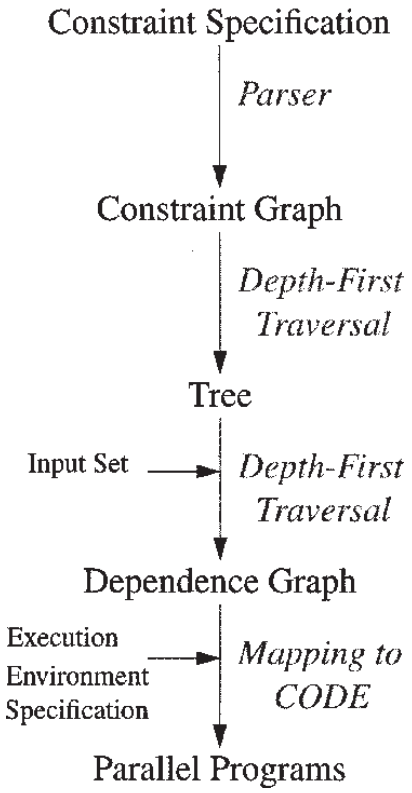
Quadratic Equation Solver

$$ax^2 + bx + c == 0, \quad 2ax = -b \pm \sqrt{b^2 - 4ac}$$

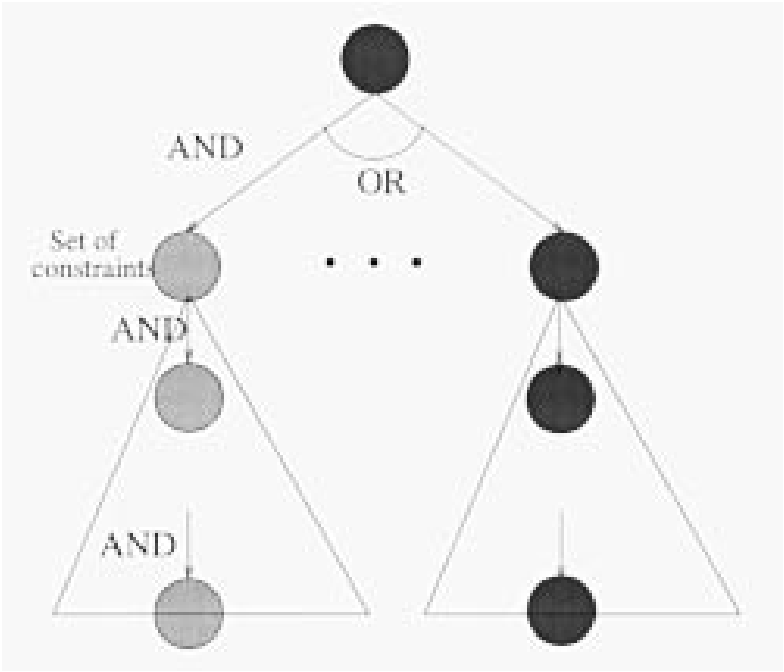
```
/* Main */
    a == 0  AND   r1 == r2   AND  b*r1 + c == 0

    OR

    a != 0 AND DefinedRoots(a, b, c, r1, r2)
```

```
/* Constraint module */

 DefinedRoots(a, b, c, r1, r2)
    t == sqr(b)- 4*a*c
    AND t>=0
    AND 2*a*r1==-b+sqrt(abs(t))
    AND 2*a*r2==-(b+sqrt(abs((t))))
```

Compilation

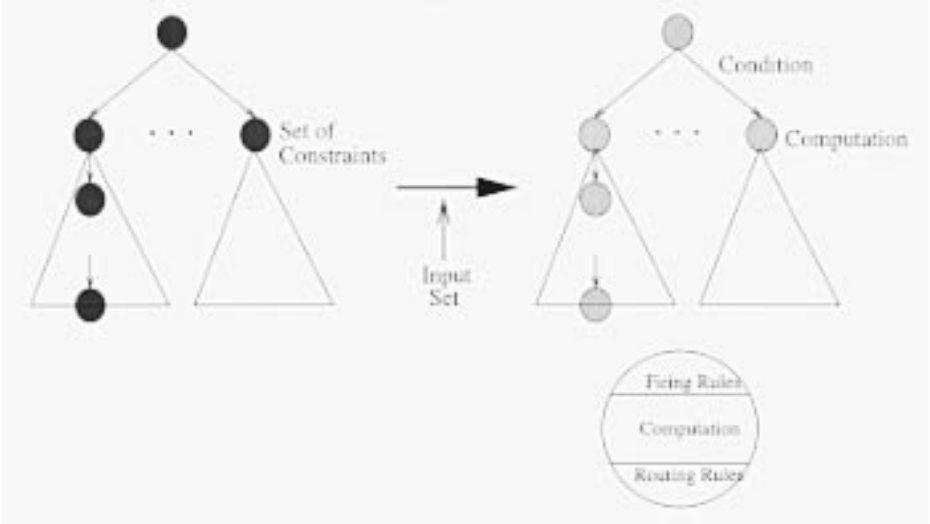


Tree from Depth-First Traversal (1)



Each path may yield a solution.

Depth-first Traversal (2)
Tree ------> Dependence Graph



Dependence Graph

Nodes are computational units.

Nodes execute dependent on certain *conditions* and the availability of data.

Resolution of Constraint Modules

Constraint Modules can be compiled to procedures

input and output parameters determined by the input set at the point of call

E.g. Constraint Module *DefinedRoots(a, b, c, r1, r2)* can be compiled to

three procedures:

- *DefinedRoots(a, b, c, r1, r2)*
- *DefinedRoots(a, b, r1, c, r2)*
- *DefinedRoots(a, b, r2, c, r1,)*

Red: Inputs, Blue: Outputs

Resolution of Constraints over Indexed Sets

Constraints over Indexed Sets can be compiled to loops over computations extracted from constraints

```
AND FOR ( i  1  10 ) {
      A[i]  +  A[i-1]   ==   0,
      A[i-1] - B[i]     ==   2
}

Input A[0]
```



```
for ( i=1;   i<=10;  i++) {
      A[i]  =  -A[i-1];
      B[i]  =    A[i-1] - 2;
}
```

Extraction of Parallelism

Types of Parallelism

AND-OR

Task

Data

Parallelism to be exploited:

1. in different paths of dependence graph (OR, Task)
2. between computations at a node (AND, Task)
3. in operations within a computation (Task)

$$\underline{M_1 * M_2} + \underline{M_1 * M_3}$$

4. in iterations of loops (Data)

Implemented

5. in primitive operations over base types (Data)

Future

Control over Granularity

Modules

Functions

Hierarchical Types

Block Odd-Even Reduction Algorithm (BOER)

Solution of A * x == d, where

$$A == \begin{bmatrix} B & C & 0 & 0 & \ldots & 0 & 0 & 0 \\ C & B & C & 0 & \ldots & 0 & 0 & 0 \\ 0 & C & B & C & \ldots & 0 & 0 & 0 \\ . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & 0 & & C & B & C \\ 0 & 0 & 0 & 0 & & 0 & C & B \end{bmatrix}$$

B, C are block matrices of order n >= 2

A is of order M*n, where $M = 2^k-1$, k >= 2

Constraint Program for BOER

1 → $BP[k-1] * x[2^{k-1}] == dP[2^{k-1}][k-1]$

AND

2 → AND FOR (j 1 k-1) {

$2 * CP[j-1] * CP[j-1] == BP[j] + BP[j-1] * BP[j-1]$

$CP[j-1] * CP[j-1] - CP[j] == 0$

AND FOR (i 0 2^{k-j}-2) {

$CP[j-1] * (dP[f_1(i,j)][j-1] + dP[f_2(i,j)][j-1])$

$== dP[i*2^j][j] + BP[j-1] * dP[i*2^j][j-1]$ }}

AND

3 → AND FOR (j k-1 1) {

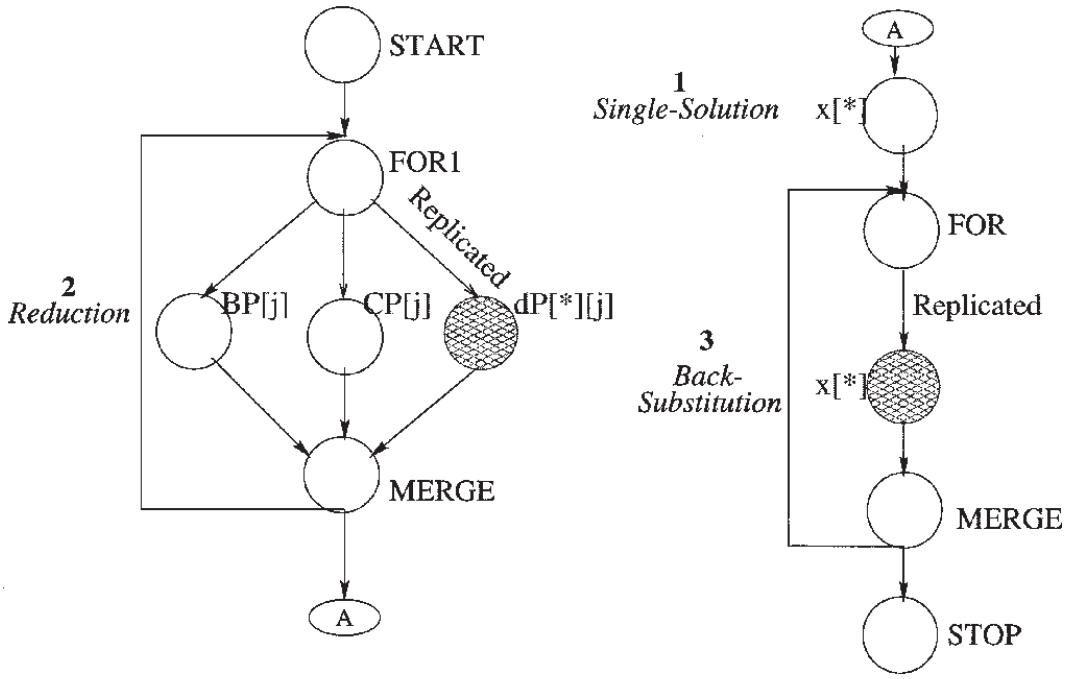
AND FOR (i 0 2^{k-j}-1) {

$CP[j-1] * (x[(i+1)*2^j] + x[i*2^j])$

$== dP[f_3(i,j)][j-1] - BP[j-1] * x[(i+1)*2^j-2^{j-1}]$ }}

Inputs=
{ BP[0],
CP[0],
dP[0][*]
}

Dependence Graph for BOER Algorithm



Summary

Constraint systems & appropriate sets of input variables

↓

generalized dependence graphs

↓

efficient parallel programs.

Granularity is controlled through
- modules,
- functions,
- hierarchical types.

Both task and data parallelism can be targeted.

Constraints:
attractive choice for a representation for problem specification

Why ?

Declarative specification

Compiler has great freedom in the generation of efficient parallel structures

Extraction of different programs from same specification

Generation of complete or effective programs

Conclusion

Feasibility of compilation of efficient parallel procedural programs from constraint programs has been established.

Future Work

Short Term

- Incorporate powerful data partitioning mechanisms.
- Complete development of an execution environment specification.
- Relax restrictions on indexed sets

Long Term

- Extraction of algorithms for applications where algorithms specification is difficult.
- How to write "good" specifications ?
- Which path should be selected in effective programs ?
- Investigate other application domains.
- Applications:
    1. Boundary Matching Problems for Decomposed Domains
    2. Data Mediation
    3. Back Tracing for Circuits
    4. Development of Distributed Network Applications

Defining the Rules of the Net: The Visions and Lessons of Self-Governance

Jeffrey B. Ritter
Director, ECLIPS
Ohio Supercomputer Center
www.osc.edu/eclips

Defining the Rules of the Net

- The compelling force of greed
- The architecture of rule-making for the Net
- The visions and lessons of self-governance (so far)
- Recommended strategies

The Compelling Force of Greed-Why we make rules

- Protection of Property Assets
- Protection of Competitive Advantage
- Protection of Communities

Realities of the Net

- Networks, including the Net, are defined by their rules.
  - technical rules
  - process rules
  - dispute resolution

Realities of the Net

- Networks, including the Net, are defined by their rules.
- Information is emerging as a new species of property.

Realities of the Net

- Networks, including the Net, are defined by their rules.
- Information is emerging as a new species of property.

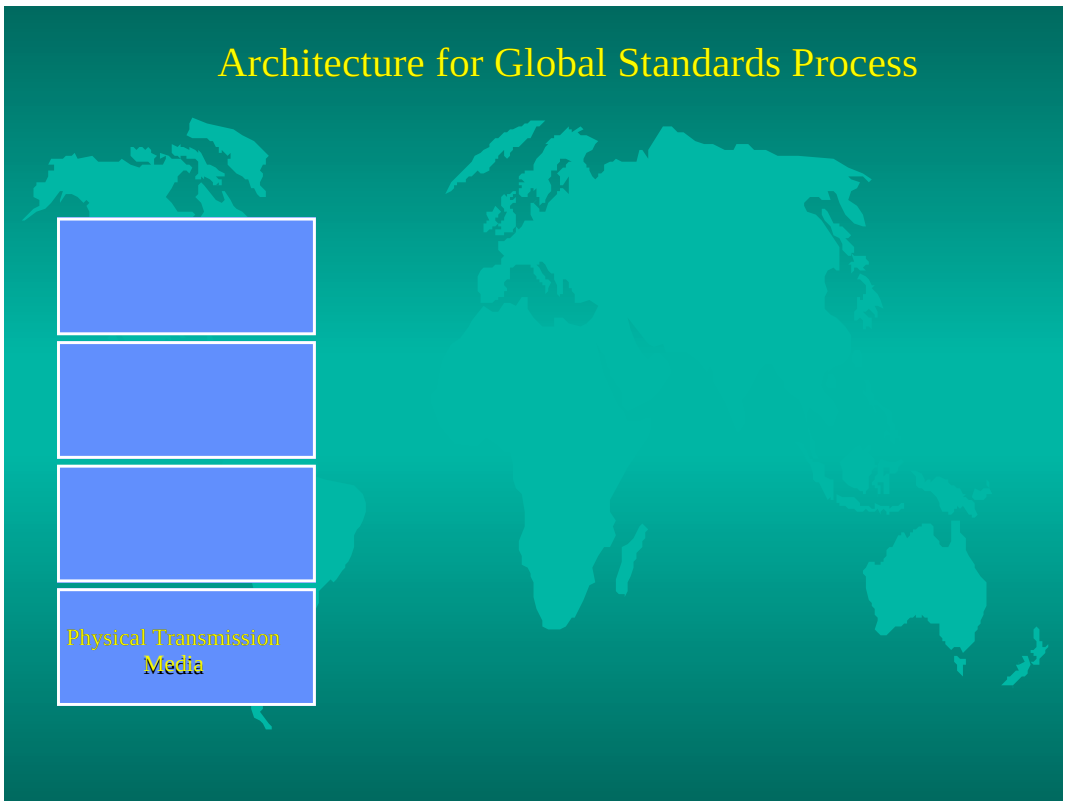
We will create rules for the Net
to protect the value of
our information assets.

How will we make the rules?

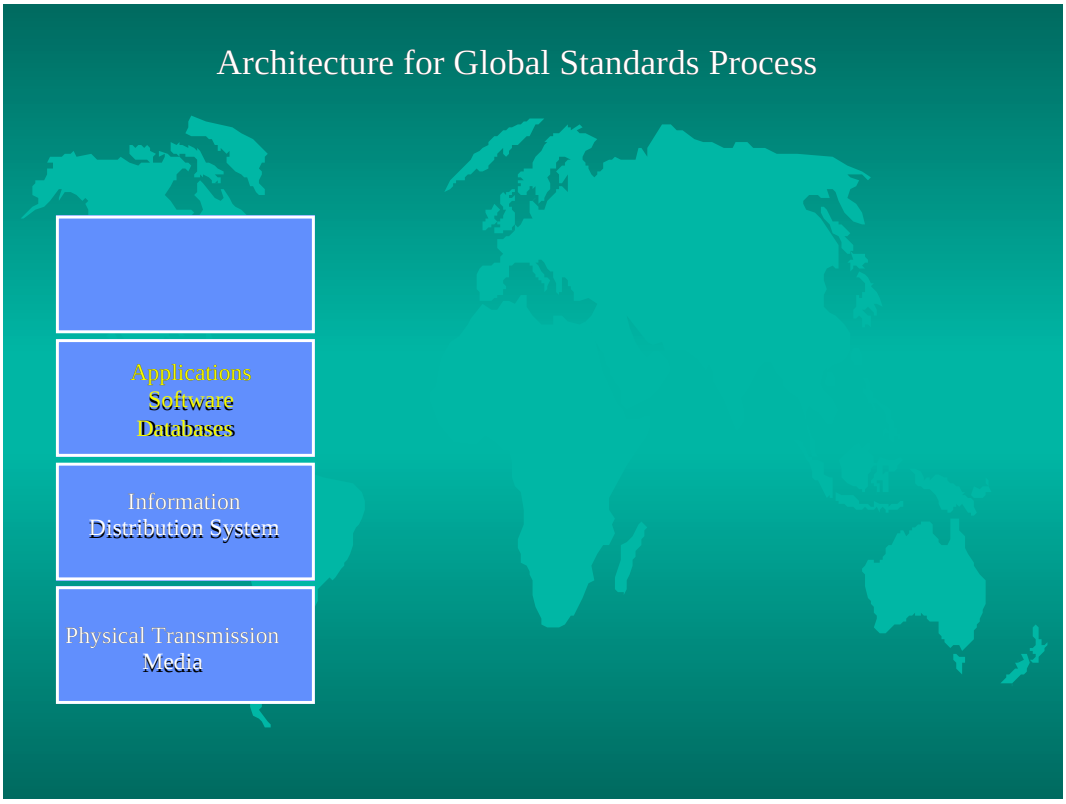
- Who is “we”?
- For what space?
- What are rules?

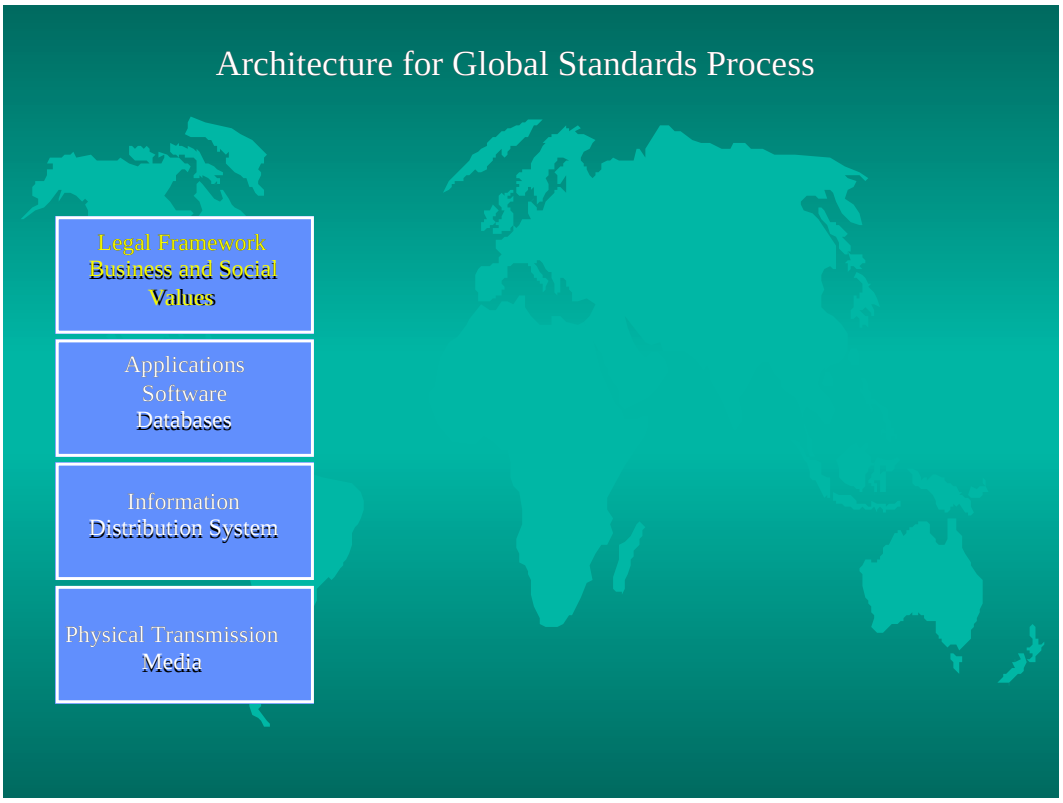
Rules--Controlling Assumptions

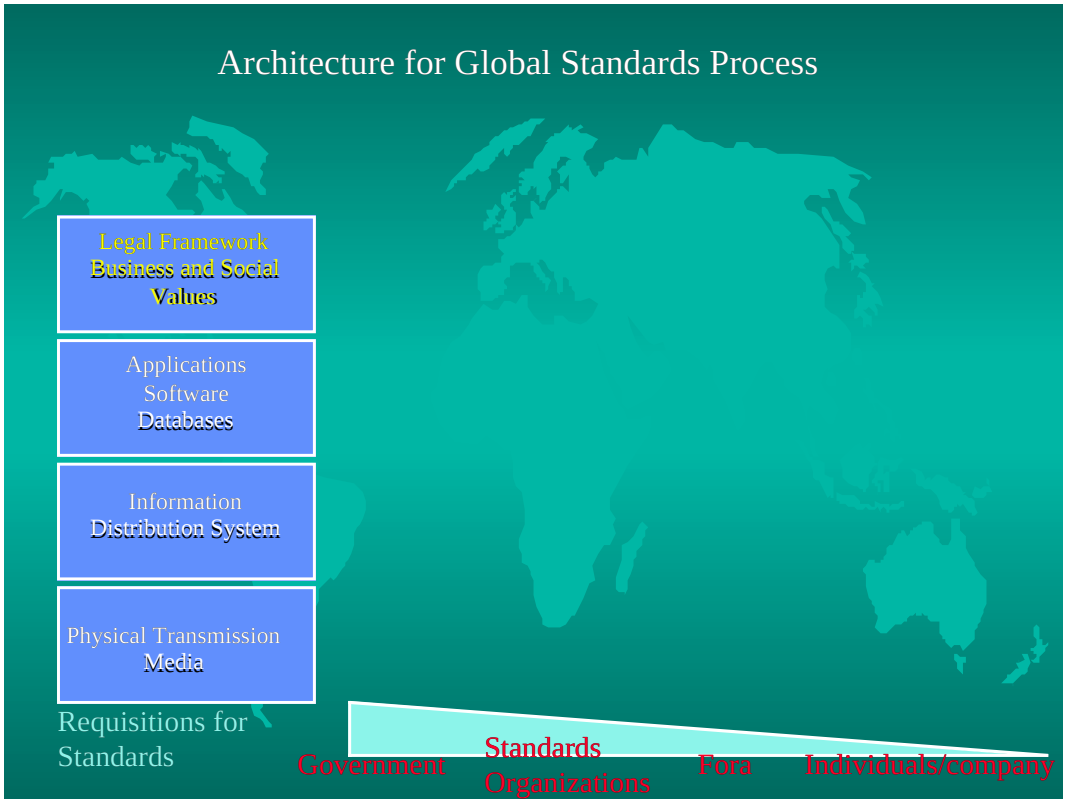
- He who makes the rules wins.
- To be free, we must regulate the Net.
- The rules will be driven by the commercial requirement for adequate returns on investment.

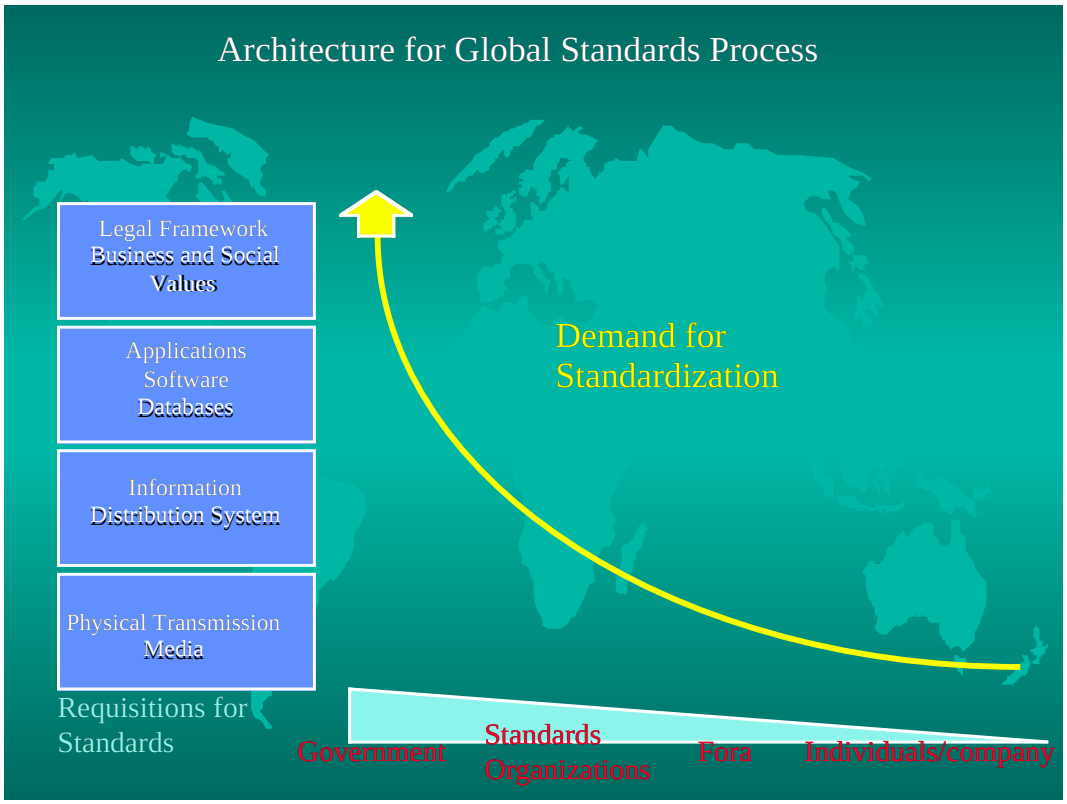

Architecture for Global Standards Process
Physical Transmission
Media

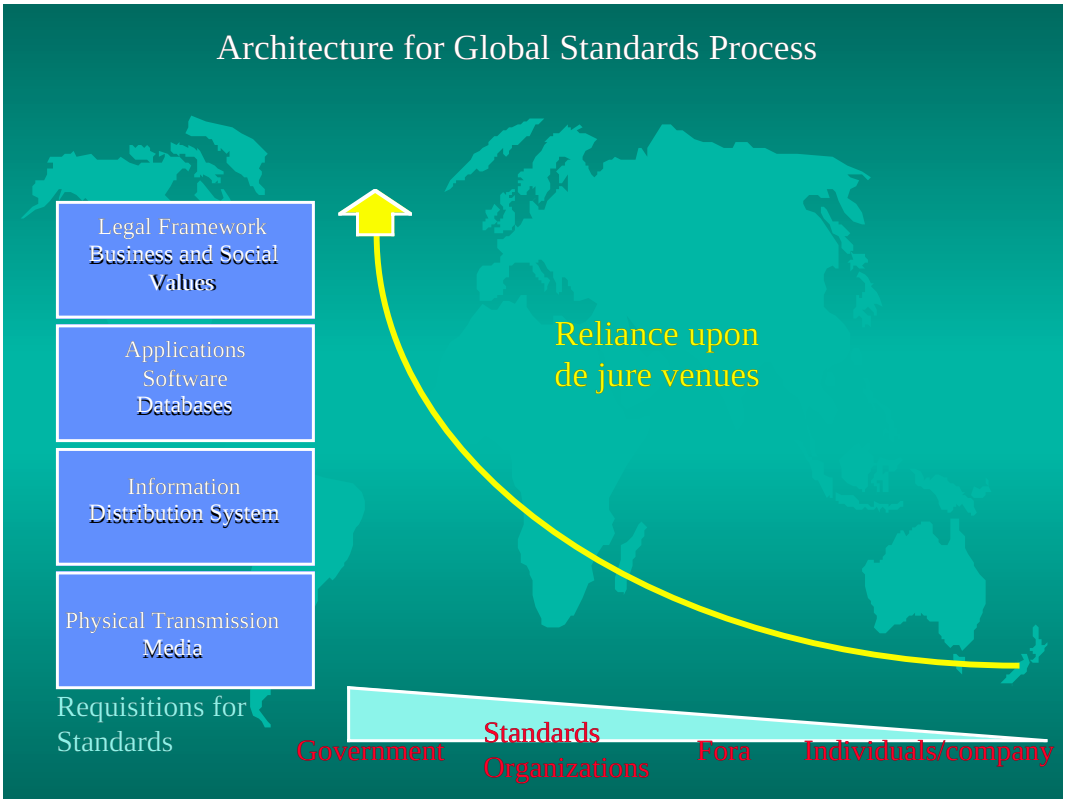

Architecture for Global Standards Process
Information
Distribution System
Physical Transmission
Media

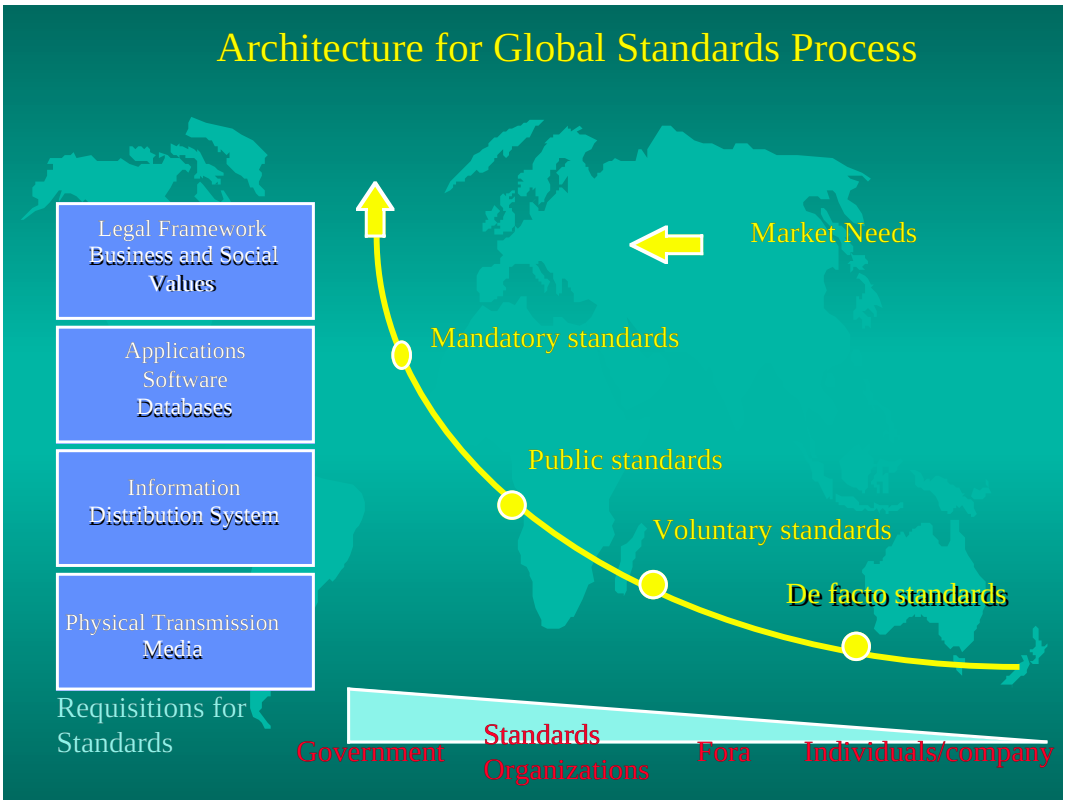

Architecture for Global Standards Process
Applications
Software
Databases
Information
Distribution System
Physical Transmission
Media

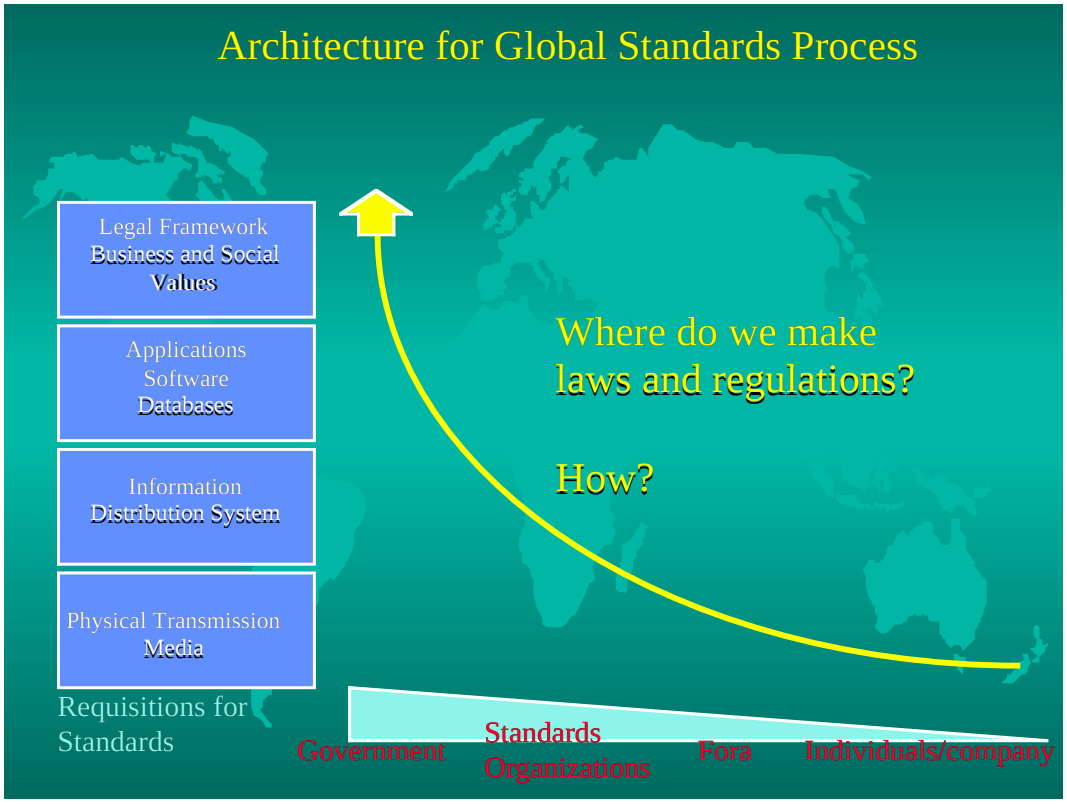

Architecture for Global Standards Process
Legal Framework
Business and Social
Values
Applications
Software
Databases
Information
Distribution System
Physical Transmission
Media

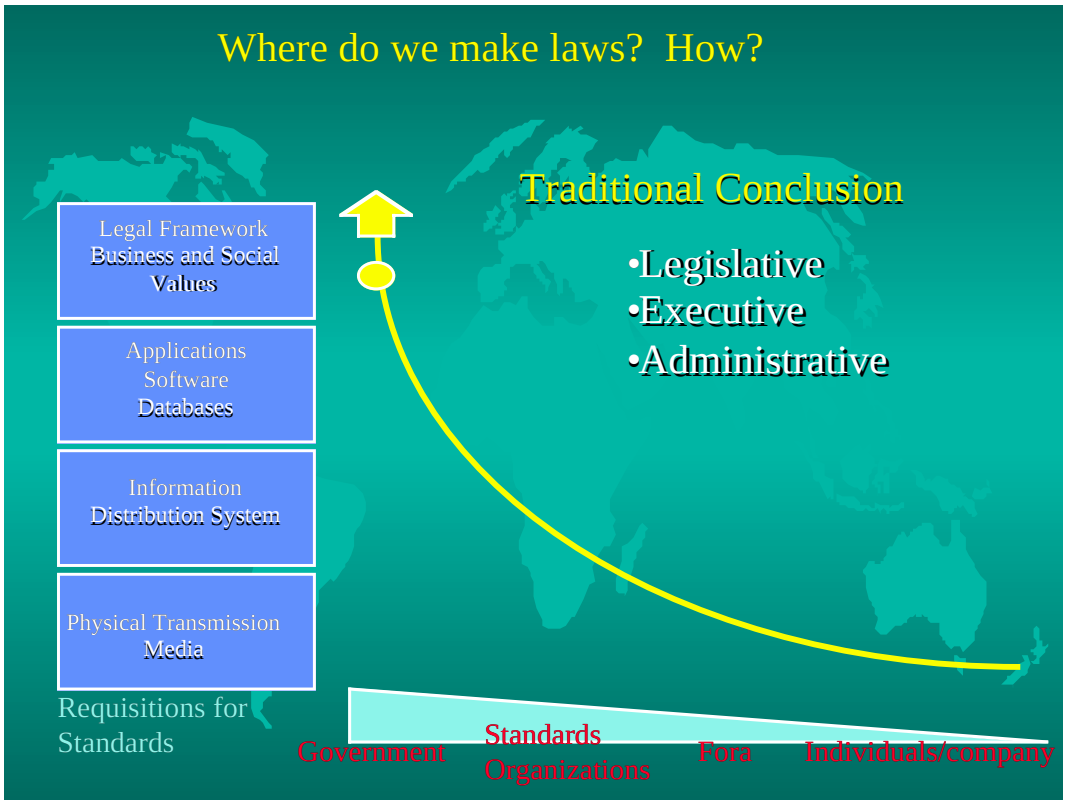

Architecture for Global Standards Process
Legal Framework
Business and Social
Values
Applications
Software
Databases
Information
Distribution System
Physical Transmission
Media
Requisitions for
Standards
Government
Standards
Organizations
Fora
Individuals/company

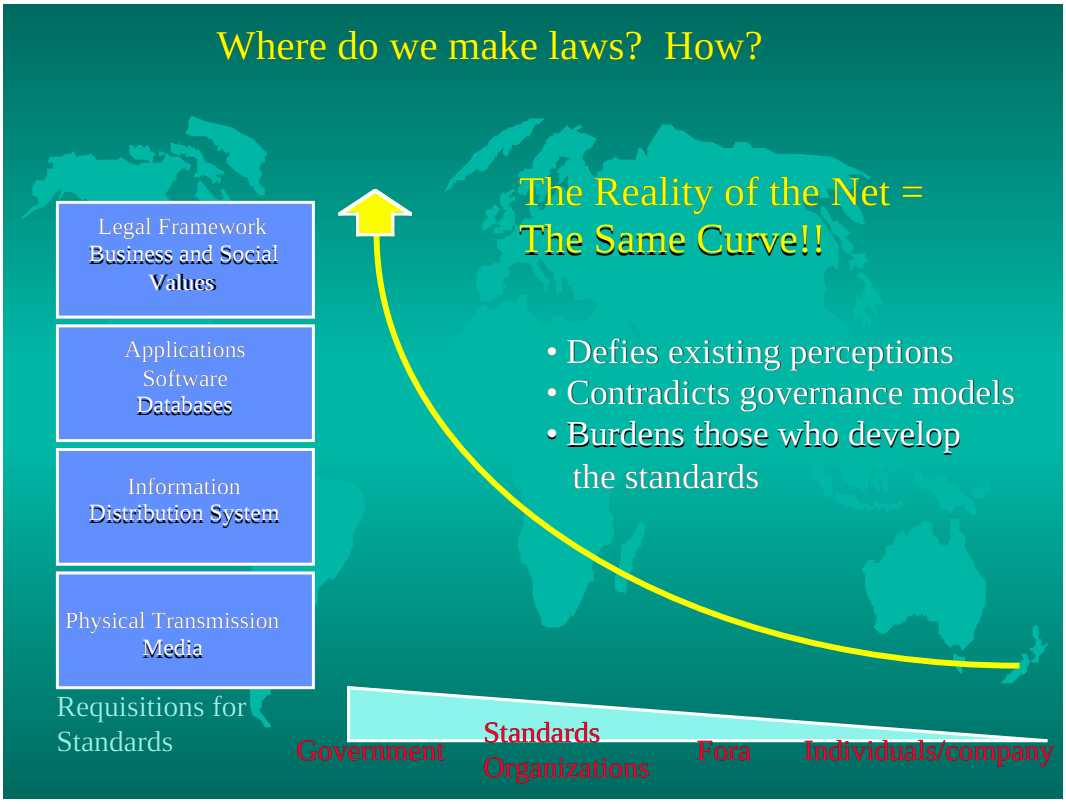

Architecture for Global Standards Process
Legal Framework
Business and Social
Values
Applications
Software
Databases
Information
Distribution System
Physical Transmission
Media
Demand for
Standardization
Requisitions for
Standards
Government
Standards
Organizations
Fora
Individuals/company


Architecture for Global Standards Process
Legal Framework
Business and Social
Values
Applications
Software
Databases
Information
Distribution System
Physical Transmission
Media
Reliance upon
de jure venues
Requisitions for
Standards
Government
Standards
Organizations
Fora
Individuals/company


Architecture for Global Standards Process
Legal Framework
Business and Social
Values
Applications
Software
Databases
Information
Distribution System
Physical Transmission
Media
Market Needs
Mandatory standards
Public standards
Voluntary standards
De facto standards
Requisitions for
Standards
Government
Standards
Organizations
Fora
Individuals/company


Architecture for Global Standards Process
Legal Framework
Business and Social
Values
Applications
Software
Databases
Information
Distribution System
Physical Transmission
Media
Where do we make
laws and regulations?
How?
Requisitions for
Standards
Government
Standards
Organizations
Fora
Individuals/company


Where do we make laws? How?
Legal Framework
Business and Social
Values
Applications
Software
Databases
Information
Distribution System
Physical Transmission
Media
Traditional Conclusion
•Legislative
•Executive
•Administrative
Requisitions for
Standards
Government
Standards
Organizations
Fora
Individuals/company



Some difficult realities

- Unlike the sea or space, the Net directly challenges the vulnerabilities of constituents for which government is often the only champion:
  - consumers
  - children/students/education
  - small to medium-sized enterprises

Some difficult realities

- Governments (fueled by the press) overreact to "market failure" in standards development
- Governments retain "veto power" over the results of self-regulation
- Standards development (including self-regulation) is capital intensive

The Consequences for Law-Making

- No "safe-harbors" readily provided for the results.
- No predictability for returns on investments.
- Reactive vs. pro-active attitudes among management and developers.

The Consequences for Law-Making

- No "safe-harbors" readily provided for the results.
- No predictability for returns on investments.
- Reactive vs. pro-active attitudes among management and developers.

Diminished realization of returns and extraordinary costs for acquiring proprietary advantages in the final layer of the standards infrastructure.

Essential Strategies

- Must encourage governments to defer the exercise of inherent jurisdiction
- Must empower standards development to embrace the complete architecture
- Must deliver methodologies for developing accelerated solutions for implementation

Deferral of Jurisdiction

- For electronic commerce, regulatory and executive sectors are prepared to defer:
  - US--Magaziner paper
  - Japan
  - European Union
  - Australia (content regulation)

Embrace the full architecture

- The gap between law and technology will diminish (and is diminishing).
- Law and policy venues are committing to technology-informed processes.
- Private sector is experimenting with self-governance.

Deliver the Methodologies of Governance

- Must develop and disseminate effective models for solutions process:
  - market-driven
  - agile
  - inclusive
  - global

Deliver the Methodologies of Governance

- Must embrace an international view--government must be considered in the process and brought to the table.
  - technology standards have consequences for national and international policies.

Deliver the Methodologies of Governance

- Must empower the value of informed choices among:
  - model agreements
  - mniform codes of conduct
  - statements of best practices
  - icon-based community-entry
- Must promote ‘safe harbors’ for deferral of government jurisdiction.

Some portending consequences

- Private sector funding of rulemaking will enhance TNO’s challenge to the nation-state.
- Bruce Sterling was right--the private sector will design enforcement power into its rules (and the new technologies).
- Diversity will undermine the consensus required for maximum security effectiveness.

Some portending consequences

The U.S. will not be able to control the policy momentum of Europe and Asia, putting at risk:

- commercial interests
- military security and management
- privacy of the individual

Privacy in the Digital Age

Deirdre Mulligan

At its core, the Digital Age represents a dramatic shift in computing and communication power. The decentralized, open nature of the network coupled, with an emphasis on user control over information, are central to achieving the First Amendment potential of the Internet. Through interactive technology, individuals today can enjoy a heretofore unknown ability to exercise First Amendment freedoms. Access to the Internet empowers individuals with an enormous capacity to speak and be heard, and listen and learn. The development of filtering and blocking devices that empower individuals to control the inflow of information gives new meaning to the core First Amendment principle that individuals should determine the ideas and beliefs deserving of expression, consideration and adherence.[1]

However, at this moment the impact of the Digital Age on individual privacy remains an open question. Will the Digital Age be a period in which individuals lose all control over personal information? Or does the Digital Age offer a renewed opportunity for privacy? The development of technologies that empower individuals to control the collection and use of personal information and communications - such as encryption, and anonymous remailers, web browsers and payment systems - are inspiring examples of the privacy-enhancing possibilities of interactive technology. However, we believe that the architecture of the Internet must be designed to advance individual privacy by facilitating individual control over personal information.

The rise of technologies that empower users of interactive communications media to affirmatively express control over personal information can fundamentally shift the balance of power between the individual and those seeking information. CDT believes this technological shift is possible and necessary, and offers us an unprecedented opportunity to advance individual privacy. However, this shift will occur if interactive media is harnessed to advance individual privacy.

Rather than responding to the very real risks posed by new technology with the Luddite-call of “smash the machine,” we are calling for a reversal of the technological status quo by demanding that technology be designed to empower people. We should seize the opportunity to vest individuals with the information and tools to express their desire for privacy in clear and effective ways, and by having those desires acknowledged by information users, we can advance privacy. We believe that this post-Luddite approach will reinvigorate individual privacy in the Digital Age.

While strengthening existing laws, such as the Fair Credit Reporting Act and the Right to Financial Privacy Act, and enacting legislation to protect health records, are crucial to protecting individual privacy, individual empowerment technologies offer a powerful method of implementing the core principle of individual control where current gaps and weaknesses leave individual privacy vulnerable. We believe that user controlled technologies that enable individuals to protect the privacy of their communications and personal information, offer an unprecedented opportunity to extend real protections for individual privacy around the world.

[1]See Turner Broadcasting Syste. Inc. v. FCC, 114 S. Ct. 2445, 2458 (1994)

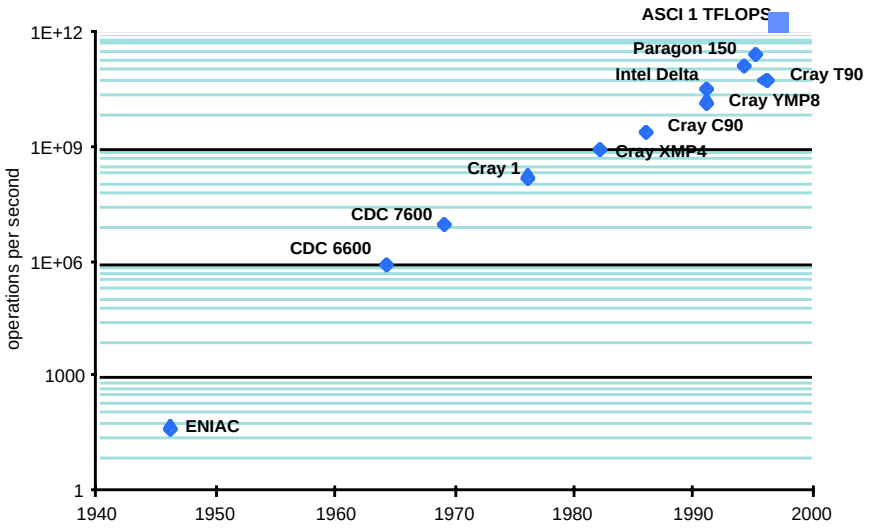
The TFLOPS Era is Here

Art Hale

Sandia National Laboratories

April 24, 1997

TFLOPS Culminates 50 Years of Supercomputing



In The Beginning of MPP

"Parallel machines are hard to program and we should make them even harder – to keep the riff-raff off them." *... an expert*

"Conversion of any code to parallel takes a few weeks, perhaps longer." *... a manager*

"Massively Parallel machines are generally considered to be hard to program special-purpose engines, but the nCUBE is different." *... a salesman*

What's the difference between a computer salesman and a used-car salesman?

The used-car salesman knows when he's lying."

... anonymous

Heyday of MPP Competition

On speed:

"The nCUBE 2 supercomputers are the fastest computing systems available today." *nCUBE brochure, 11/91*

"The Touchstone Delta system is the world's fastest production computer." *Intel brochure, 11/91*

On Popularity:

"CM systems are the most popular parallel supercomputers in the world." *TMC document, 11/91*

"The iPSC/860 has become the most widely used parallel supercomputer in the world." *Intel brochure, 11/91*

On the Future:

"The CM-5 incorporates the first parallel supercomputer architecture that scales to TeraFlops." *TMC document, 11/91*

"Intel is the first to offer a proven technology path to take you to TeraFlops." *Intel brochure, 11/91*

There are three rules for programming parallel computers.

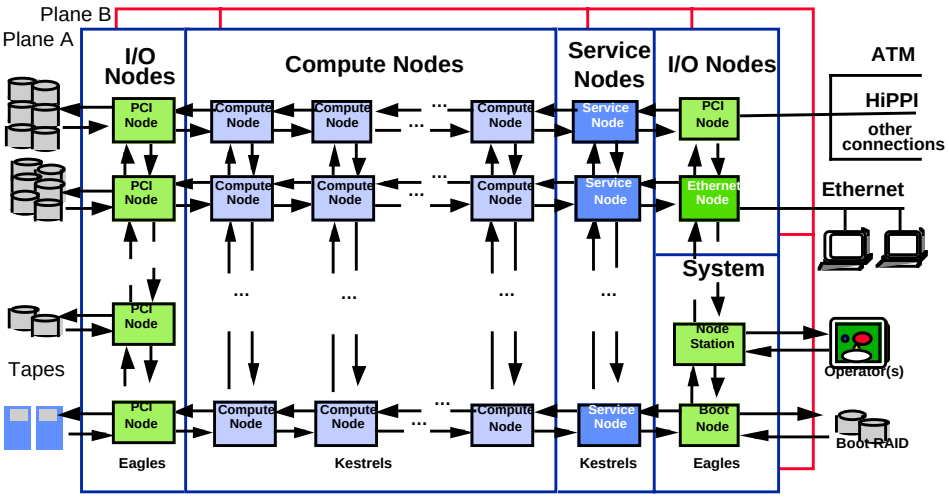
We just don't know what they are yet.

Gary Montry

Rules of Parallel Programming?

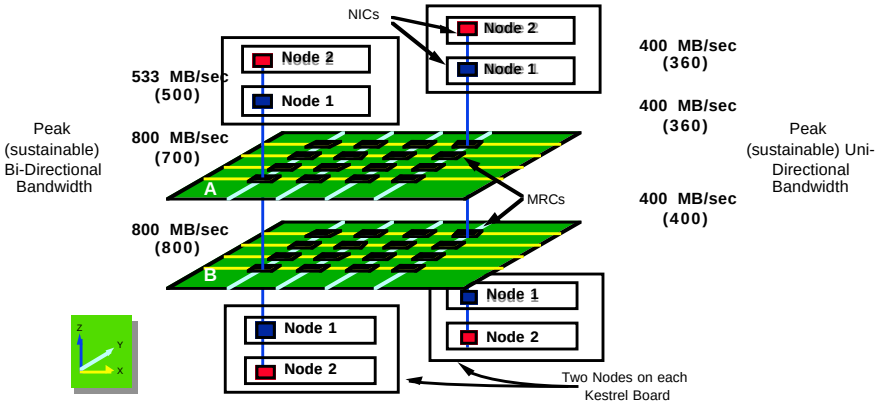
1) *Focus on Scalability*
2) *Refer to Rule #1*
3) *Refer to Rule #2*

Integrate Distributed Computing

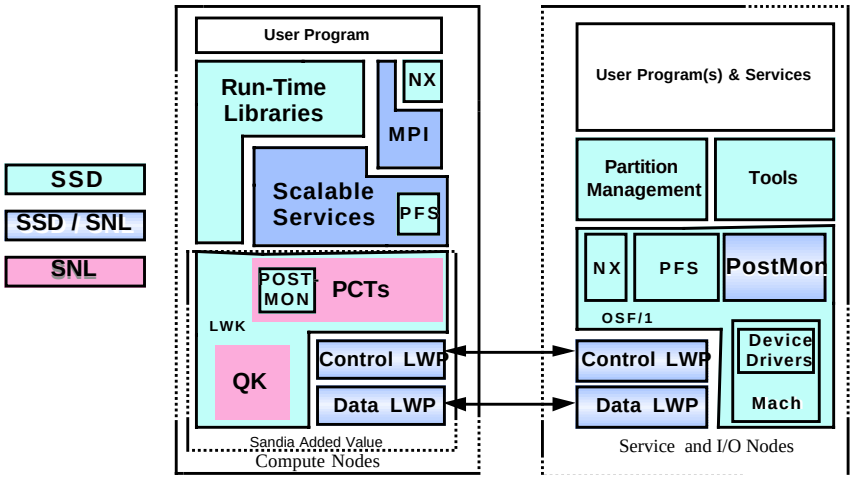


Specialized servers and services
Closely integrated "system image"

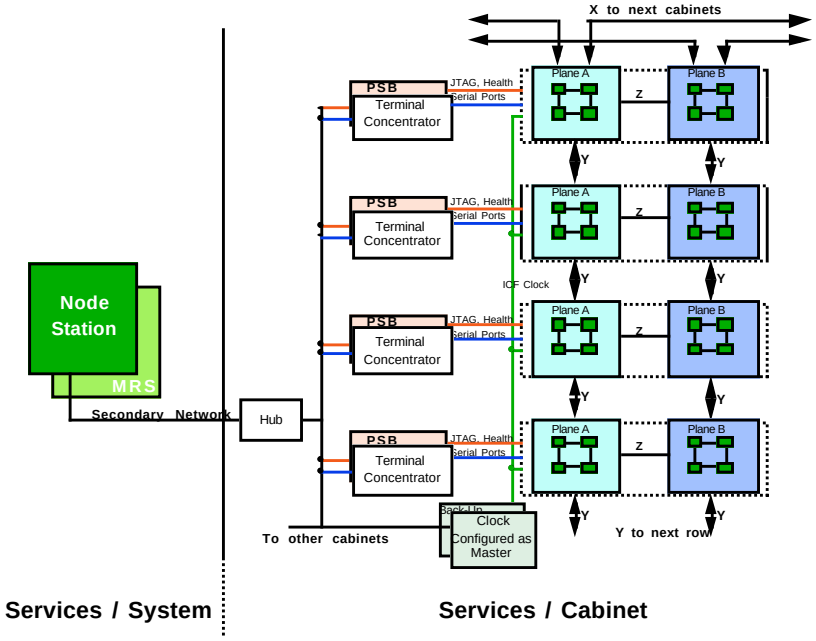
Balance Communication/Computation



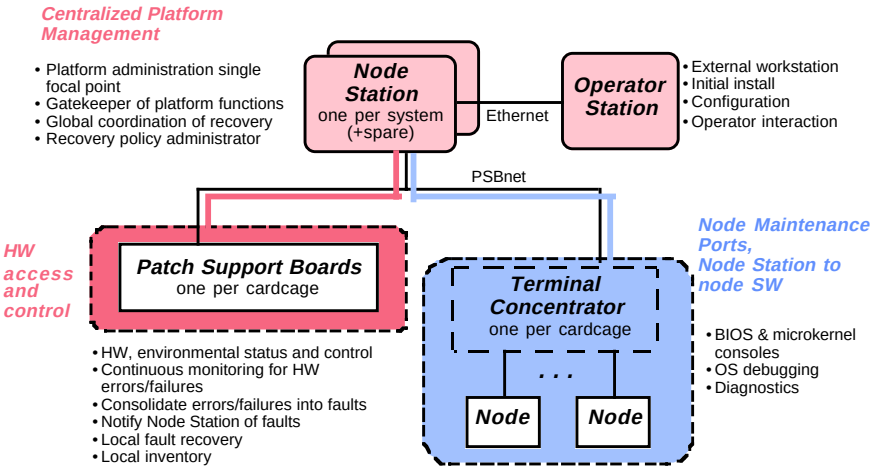
Scalable Operating System



Build In Scalable RAS Features



Scalable Administration



Scale Applications

Materials

- electronic structure (BANDPW)
- quantum chemistry (QUEST)
- MD for molecules (ParBond)
- MD for metals (ParaDyn)
- grain growth evolution (PGG)
- gas separation and adsorption
- GCMD for diffusion
- Quantum Monte Carlo
- Hartree Fock Methods
- electron microscopy (PMC)

Signal and Image Processing

- SAR image simulation (SRIM)
- radar signatures (XPATCH)
- phase gradient autofocus (PGA)
- seismic modeling
- acoustic beam formation for ASW
- magnetic resonance imaging

Engineering Simulations

- shock physics (PCTH)
- low density flows (DSMC)
- chemically reacting flows (SALSA)
- structural mechanics
- integrated shock/structures (ALEGRA)
- combustion
- inverse scattering for semiconductor manufacturing
- ocean modeling
- electromagnetism
- aerodynamics
- integrated thermal/solid mechanics (DELTA)

Information Processing

- cryptanalysis
- tracking and correlating

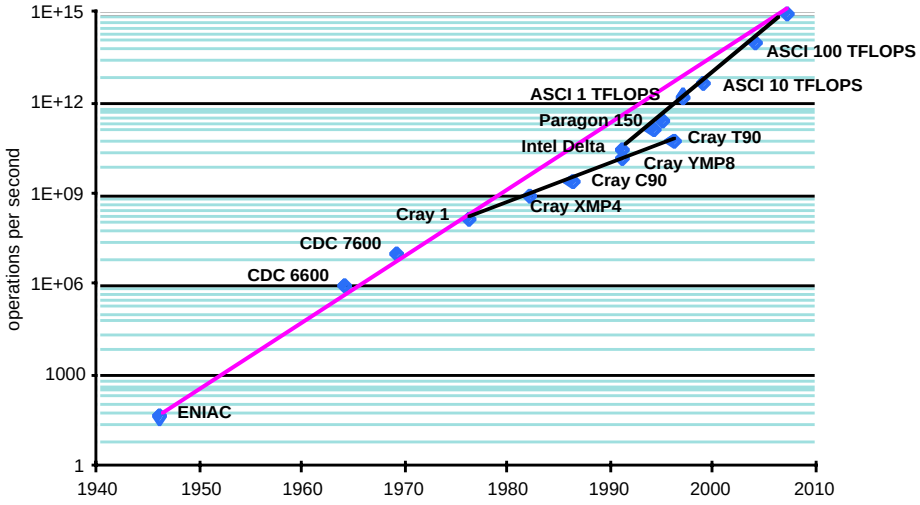
"When you come to a fork in the road – take it."

Yogi Berra

Some Future Challenges

- **Unbundled building blocks**
  - **nodes, interconnect technologies, system software, programming environment**
- **Load balancing and heterogeneity**
- **Abstract system views**
  - **e.g. application configuration independence**
- **Automated system discovery and configuration (plug & play)**
- **Fault tolerance**
- **Simplifying system integration**

PetaFlops in 2007 is Consistent with Historical Evolution



"What is the world's fastest computer and how fast is it?

Currently, it's an HP notebook. It's used on the Space Shuttle to compute orbital position and has been clocked at 17,500 mph."

Robert Hyatt

ASCI Applications Challenges

Ken Koch
ASCI Code Developer
Applied Theoretical & Computational Physics
Los Alamos National Laboratory
krk@lanl.gov

ASCI Applications Overview

What is ASCI?

- Accelerated Strategic Computing Initiative
  - Goal
    - Computational simulation as a tool for addressing safety, reliability, and performance issues in the absence of nuclear testing
  - Structure
    - 1 Program / 3 Labs (LANL, LLNL, Sandia,DOE HQ)
    - Applications & Problem Solving Environment
    - Platform strategies
      - Red, Blue Pacific, Blue Mountain, ...(and more)
    - Alliances, Pathforward, ...

Simulation Objectives

- Physics
  - 3D highly resolved multi-material complex geometries with large deformations
  - multiple spatially-collocated coupled physical processes
- Computer Science
  - portability & standards
    - languages, APIs, distributed & shared memory
  - scalable parallel processing
    - 1 million to 1 billion mesh sizes
    - Order(1K to 10K) CPUs

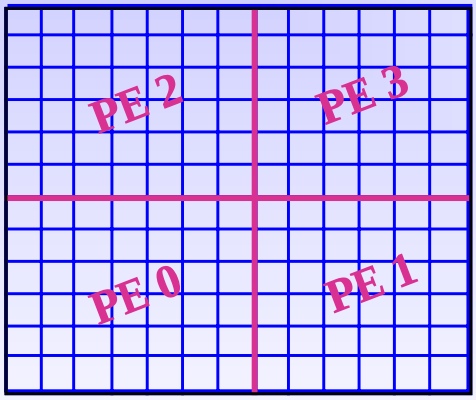
Programming Approaches

- Start with current defacto standards
  - portability & longevity
  - better leveraging through broad user-base
  - lower risk
- Building blocks
  - C++, Fortran90, & C languages
  - MPI & domain decomposition
  - add multiprocessing as 2nd step
    - a wish for good parallelizing compilers
    - within each domain or across domains

Major LANL ASCI Applications Projects

Blanca Project

- Regular structured grids



X(imax,jmax)

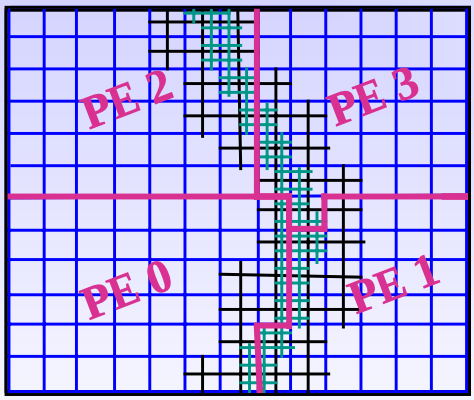
X(i,j) - X(i+1,j)

Blanca Codes

- Tecolote-2
  - multidimensional Eulerian / ALE hydro
  - POOMA C++ OO framework
    - data distribution, methods, & MPI / threads
  - multiple ordered data blocks per PE
    - domain overloading
    - variable data per cell
- Euler (quick port)
  - Fortran90 & MPI “cshifts”
  - one ordered data block per variable per PE
    - 1 or M data per cell

Crestone Project

- Regular base grid with cell-by-cell adaptive refinement



X(ncell)
Face(nface,2)
Index(nface,2,2)

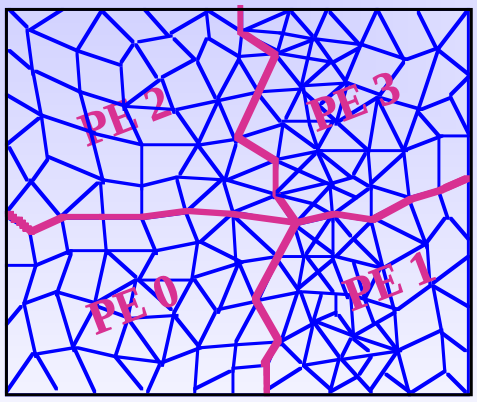
X(Index(i,LO,Xdir)) -
X(Index(i,HI,Xdir))

Crestone Codes

- Rage90
  - multidimensional Eulerian rad-hydro
  - Fortran90 with MPI communications
    - MPI global gather / scatter exchanges
  - oct-tree data stored in 1D arrays with index arrays for connections
    - arbitrary cell ordering within each PE
    - one dynamic array per “physics” variable
    - variable data can be packed per cell
    - data count changes every time step
  - export controlled

Shavano Project

- Unstructured polyhedron grids



X(imax)
Index(neigh)

X(i) - X(Index(n))

Shavano Codes

- Flag code
  - multidimensional multiple-grid hydro
  - Fortran77 & C & data storage framework
  - polyhedral finite difference meshes
  - 1D data arrays with storage manager
- Chad & Dante codes (alternate track)
  - 3D Eulerian hydro and transport
  - Fortran90 & MPI library for gather / scatter
  - hexahedral non-moving finite elements
  - 1D data arrays with index arrays

Parallel Scalability Issues: Now and in the Future

Load Balancing

- spatial decomposition at least
  - 10^4 to 10^6 cells / PE (think crudely: 20^3 to 100^3)
  - possible additional decompositions over energy, etc., but with more communication
  - possible domain overloading
- dynamic in time
  - even if the grid is static!
    - e.g. mixed materials & turbulent jets
  - everyone is moving toward adaptivity
  - work monitoring & migration necessary
  - more domains aggravate the situation

Load Balancing

- multi-physics timestep
  - inherent imbalances
    - mixed materials, regional physics, particle field vs. grid views, AMR levels
  - timestep example:
    1. global explicit physics
    2. zone-local (zero-D) region-restricted physics
    3. global implicit physics
  - different decompositions per physics?
    - more data motion / copying
    - higher communications

Sparse Matrix Solvers

- currently CG with diagonal scaling
  - easy to implement in message passing
  - it works and is robust for now
- distributed matrix-vector multiply
  - only ~2 FLOPs per communicated element
  - irregular or adaptive grids
    - on-PE memory gathers
    - off-PE communications gathers
    - variable number of coefficients per row

Sparse Matrix Solvers

- millions to billions of unknowns
  - bigger problem sizes on bigger machines
  - convergence problems
    - huge number of degrees of freedom
    - are existing methods robust enough?
    - required iterations likely to increase
  - is parallel irregular multigrid an answer?
- performance
  - time-to-solution **not** parallel speedup
  - latency problems because of limited FLOPs per communicated element

Software Development

- compilers / languages
  - C++ and Fortran90 need to mature more
  - parallelizing compilers
    - support for C++, Fortran90, & C with mixed-language compatibility
    - scalability limit & tradeoffs with MPI
  - Fortran90 + MPI
    - copy-in / copy-out + asynchronous operations
    - user-derived types
    - “choice” arguments & strong typing
    - Fortran95 & MPI-2 are not enough
  - true parallel languages or extensions

Software Development

- debuggers
  - need solid support for the latest languages and newest features
  - support for all parallel features
    - MPI
    - irregular distributed data
    - parallelizing compilers
    - threads
  - usability on large parallel systems
  - visualizations instead of prints

Software Development

- parallel I/O
  - simple methods not scalable
    - 1 file per MPI process
    - single PE I/O with MPI gather/scatter
  - need a proven widely-supported API standard and good implementations
    - parallel reads/writes to single logical files on high performance parallel filesystems
  - need for high-level self-describing files, not just fast low-level binary I/O

Better Physics

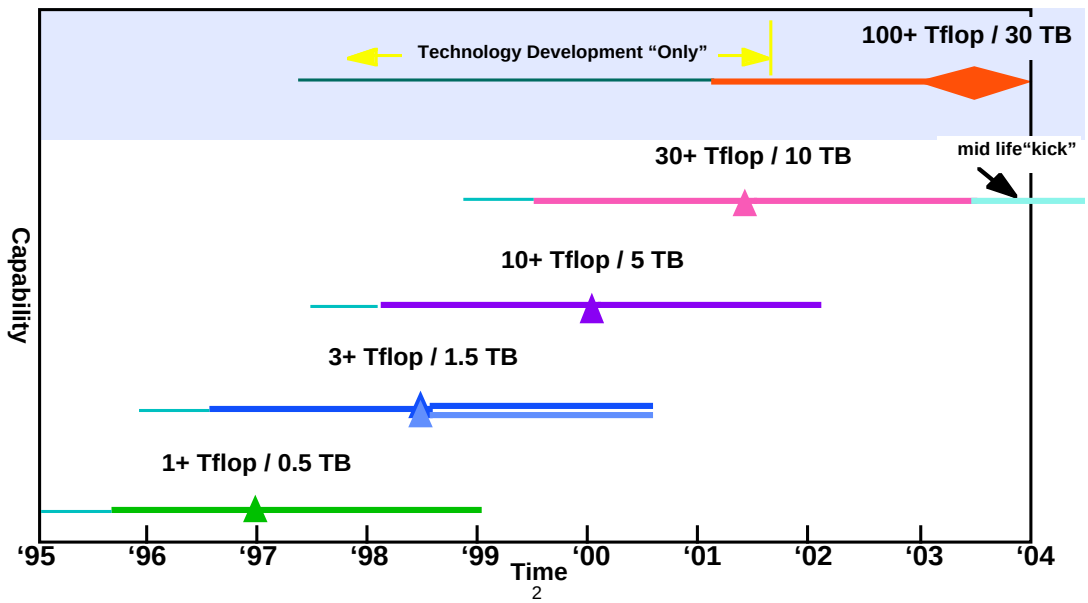
- better fidelity demands not just higher resolution but better physics as well
- more effort to develop & incorporate new physics since they need to be parallel implementations

The job is never done!

Day-to-Day Programmatic Usage of 100 TFLOP/s Systems Demands Careful Balance in the Overall Computing Environment

ASCI Session at Salishan Conference
April 23, 1997
Dr. Mark K. Seager
Asst. Department Head
Scientific Computing and Communications Department
seager@llnl.gov

ASCI Platforms Roadmap

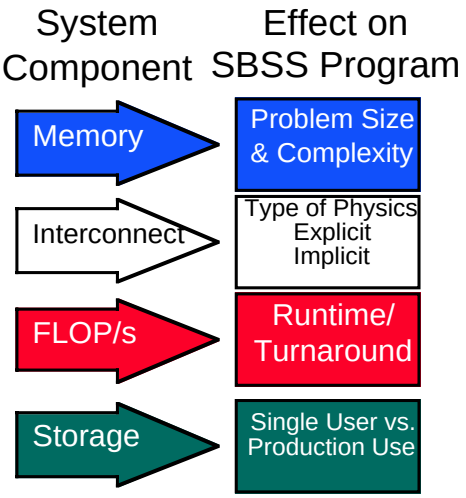


Setting for 100 TF Machine

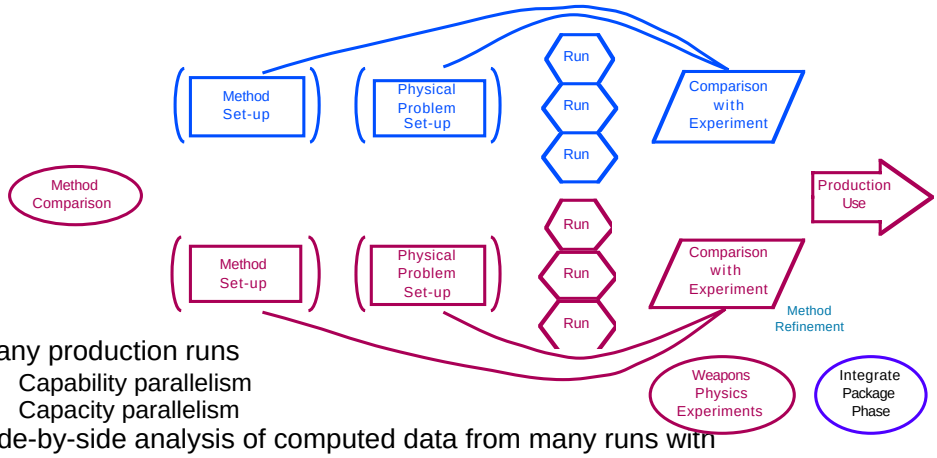
- ★ CY2004
  - » Al Gore starting second term
  - » BART establishing service to SFO
  - » Bullet Train between Sacramento and LA
- ★ Technology Trends
  - » "Processors" achieve ~8 GF performance
  - » 100 TF machine is ~12,500 processors
  - » Power about 5-10 MWatt
  - » Cooling about 1,500-2,300 Tons (3-4 MWatt power)
  - » 5,000-15,000 FT^2 floorspace

Balanced System for Science Based Stockpile Stewardship

- ★ *Mission Driven Ratios*
  - » 2.5 TB Memory
  - » 2.5 Tb/s Bi-Sectional B/W
  - » 3.0 TeraFLOP/s Peak
  - » 75 TB Disk
  - » 90 GB/s I/O B/W

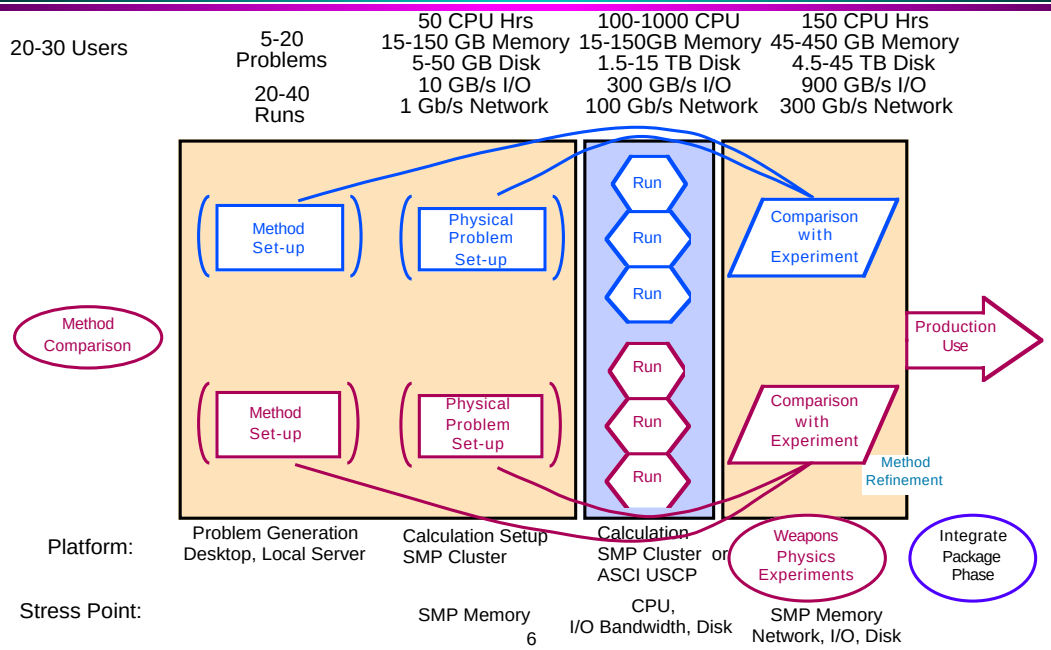


Typical Method Comparison Calculation Regime

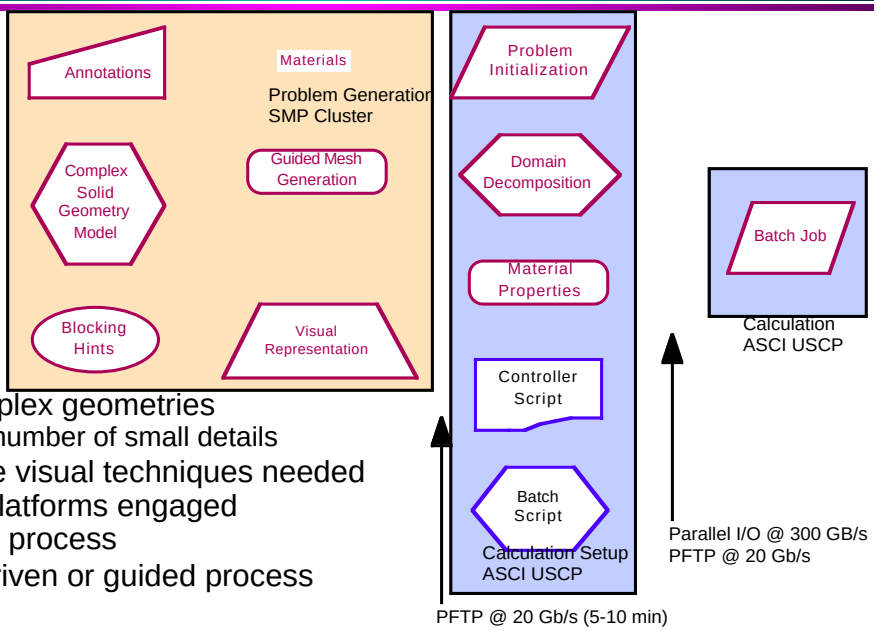


- ★ Many production runs
  - » Capability parallelism
  - » Capacity parallelism
- ★ Side-by-side analysis of computed data from many runs with experimental data
- ★ One, two and three dimensional data comparison; many involving time dependent variables
- ★ Method set-up and physical problem set-up visualization intensive and very time consuming

Typical Set-up and Production Utilize Multiple Platforms with Different Levels of Performance



Preperation for "Full-Up Calculation" is Very Labor Intensive



- ★ Very complex geometries
  - » Large number of small details
- ★ Innovative visual techniques needed
- ★ Multiple platforms engaged
- ★ Multi-step process
- ★ Human driven or guided process

Secondary "Full-Up Calculation" Scenerio

• **Specific A-Division 3-D Production Example**

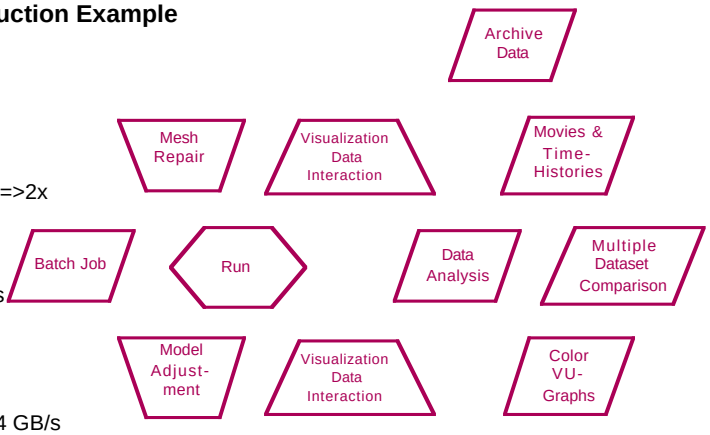
CPU
50 TFLOP/s = 80 Hrs runtime
= 1 Week wall = 10^6 CPU Hrs

Memory
30TB Memory (~20 Billion Zones)

Swapping
Hourly, 5% Overhead, Read/Write=>2x
75% total memory on machine
=> 416 GB/s I/O to local disk

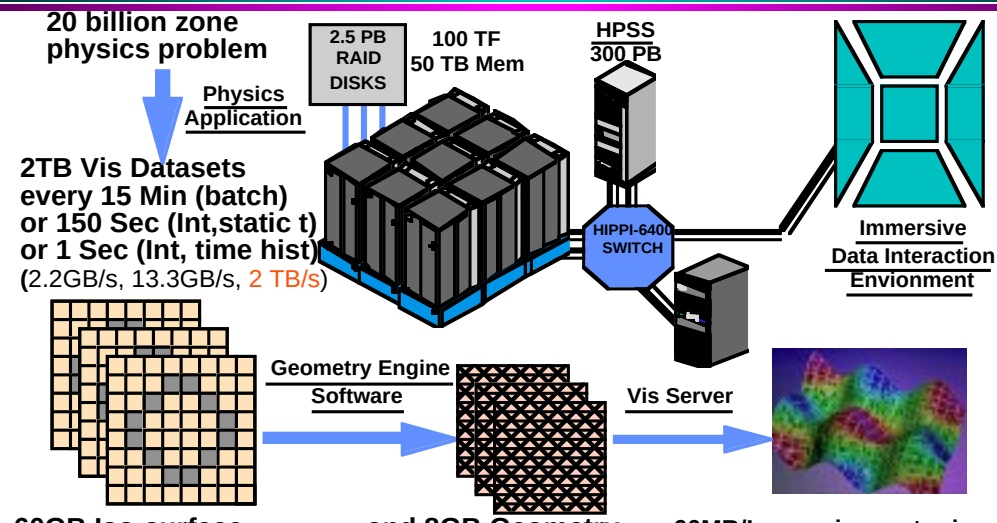
Archive
50% Job memory, 15 min intervals
=> 16.7 GB/s I/O to tape
320 restarts => 48 PB

Disk
20 BZ = 2 TB/Frame for vis
15 min intervals @ 5% sync = 40.4 GB/s
320 frames = 640 TB

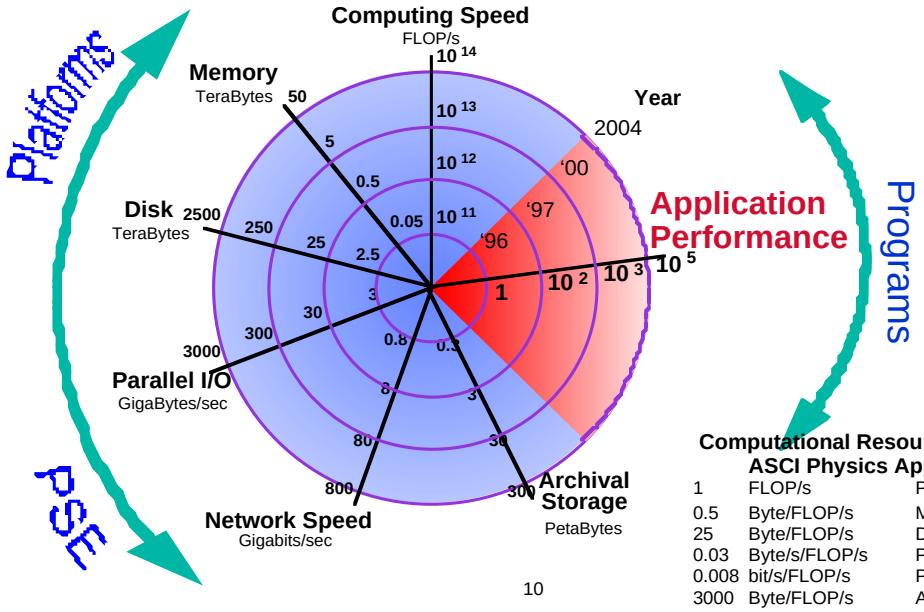


- ★ A calculation is not just a batch job. It is a series of batch jobs
- ★ Most calculations are "steared" in a batch sense.
- ★ Frequent interaction with archived data
- ★ Frequent need to visualize mesh, mesh based variables and materials

CY04 Ultra-Scale Computing for Immersive Data Interaction



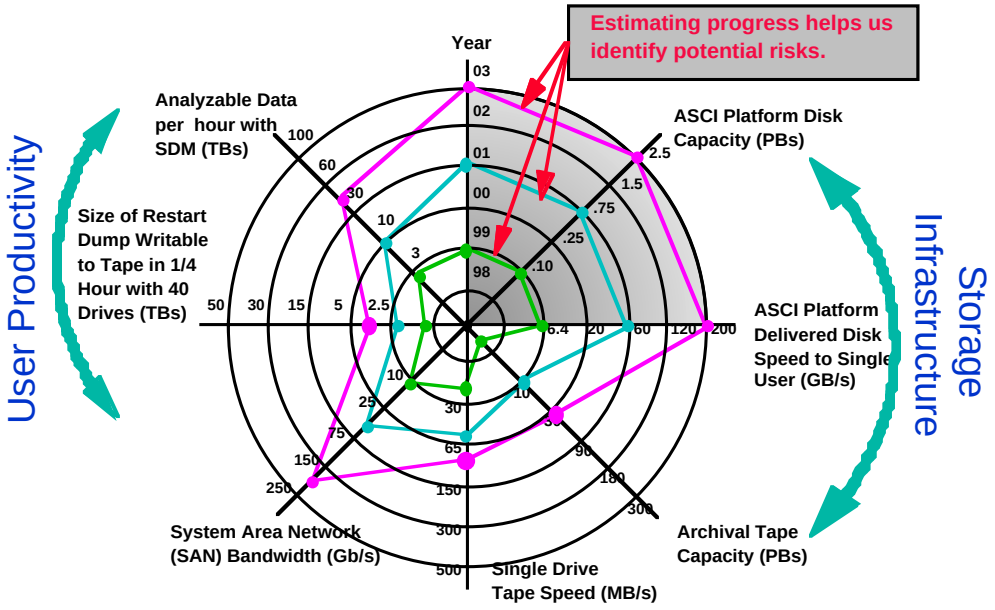
The Key to a Usable System is Application Driven Scaling



Computational Resource Scaling for ASCI Physics Applications

1	FLOP/s	Peak Compute
0.5	Byte/FLOP/s	Memory
25	Byte/FLOP/s	Disk
0.03	Byte/s/FLOP/s	Peak Parallel I/O
0.008	bit/s/FLOP/s	Peak Network
3000	Byte/FLOP/s	Archive

Designer productivity risk due to archival storage limitations



Summary

- ★ 100 TF computers will give ASCI users new capability that would otherwise be unattainable.
- ★ The computing environment around these machines must be carefully scaled up as well because this is a huge impact on the day-to-day productivity that ASCI users will experience.
- ★ Multiple classes of machines (full spectrum) are utilized at various stages in the problem set-up, calculation and analysis phases of a series of runs.
- ★ In the 100 TF computing regime, the “big-data” problem becomes the “huge-data” problem. Innovative approaches to storage, networking and visualization must be developed and deployed on a large scale.