Content Management • Enterprise Solutions • XML Labs

XML JOURNAL

THE ULTIMATE XML ENTERPRISE RESOURCE

March 2003 Volume: 4 Issue: 3

xml-journal.com

page 28 CONFERENCE PROGRAM INSIDE

$6.99US $7.99CAN
04
0 09281 01314 3

SYS-CON MEDIA

Established best practices and emerging design patterns Ayesha Malik 36

XML JOURNAL

FOUNDING EDITOR
Ajit Sagar ajit@sys-con.com

EDITORIAL ADVISORY BOARD
Graham Glass graham@themindelectric.com
Coco Jaenicke cjaenicke@attbi.com
Sean McGrath sean.mcgrath@propylon.com
Simeon Simeonov talktosim@polarisventures.com

EDITORIAL
Editor-in-Chief
Hitesh Seth hitesh@sys-con.com
Editorial Director
Jeremy Geelan jeremy@sys-con.com
Managing Editor
Jennifer Stilley jennifer@sys-con.com
Editor
Nancy Valentine nancy@sys-con.com
Associate Editors
John Evdemon jevdemon@sys-con.com
Jamie Matusow jamie@sys-con.com
Gail Schultz gail@sys-con.com
Jean Cassidy jean@sys-con.com

PRODUCTION
Production Consultant
Jim Morgan jim@sys-con.com
Art Director
Alex Botero alex@sys-con.com
Associate Art Directors
Louis F. Cuffari louis@sys-con.com
Richard Silverberg richards@sys-con.com
Assistant Art Director
Tami Beatty tami@sys-con.com

CONTRIBUTORS TO THIS ISSUE
Kristian Cibulskis, Ayesha Malik, Suhayl Masud, Eric Newcomer, Hitesh Seth, Sim Simeonov, Yuhang Sun

EDITORIAL OFFICES
SYS-CON MEDIA
135 CHESTNUT RIDGE ROAD, MONTVALE, NJ 07645
TELEPHONE: 201 802-3000 **FAX:** 201 782-9637
XML-JOURNAL (ISSN# 1534-9780)
is published monthly (12 times a year)
by SYS-CON Publications, Inc.
Periodicals postage pending
Montvale, NJ 07645 and additional mailing offices.
POSTMASTER: Send address changes to:
XML-JOURNAL, SYS-CON Publications, Inc.,
135 Chestnut Ridge Road, Montvale, NJ 07645.

SYS-CON MEDIA

Integration Matters

WRITTEN BY **SIMEON SIMEONOV**

The challenge of integrating software and systems will always be with us. In the brief but turbulent history of information technology, creation and destruction go hand in hand. Old technologies and approaches give way to new ones, sometimes quietly and sometimes with a fight. Yet, in this maelstrom of activity one thing remains unchanged. Our desire to solve bigger and more important business problems breeds increasing complexity. To battle this complexity we divide and conquer. We don't want to reinvent the wheel. We try to wrestle control of problems that we know how to solve and then we build up from there to a final solution. The larger pieces become independent systems and applications. Integration – broadly speaking, the superset of EAI, B2B, BPM, and many other acronyms – is the process through which the larger pieces work together to solve problems.

In my article in this issue,"Integration Is the Killer App," I argue exactly what the title claims. Integration is how we extract value out of systems above and beyond what they were originally designed for. In a connected world, where more and more applications and data sources are available both within and beyond the firewall, integration is the key to managing information and leveraging network effects.

Integration would be easy if only the problems we wanted to solve had many similarities and if the methodologies and tools for decomposition and recomposition were well developed. Of course, this is not the case. In Eric Newcomer's words "the software industry remains essentially a craft business." The problems we are trying to solve are very diverse, even within vertical domains. We reinvent the wheel every day, multiple times. We cannot agree on methodology, perhaps because we have not developed a good one yet. Well-trained integration personnel are scarce, even in this economy. Therefore, we have to put our hope for making integration easier and more efficient in the development of integration technologies and best practices.

XML and Web services are our current best hope for lowering costs across many of the facets of integration.

This month, ***XML-J*** addresses a number of important areas for integration. In "Can Software Cross the Standards Divide?", Eric Newcomer looks at the relationship between standards, competition, and innovation. Ayesha Malik explores XBRL, an emerging standard for expressing business information, in her new monthly column addressing standards activity. While it may be easy to standardize on the basic mechanisms for exchanging messages between applications, it is not always easy to have meaningful interactions. One way to add meaning is to agree on vocabularies. Kristian Cibulski's article, "An Introduction to BSML," introduces the Bioinformatic Sequence Markup Language (BSML), a standard mechanism to communicate genomics research information. When two applications cannot speak the same language, translation becomes necessary. XSLT can transform XML documents from one vocabulary to another. Yuhang Sun's article, "Developing Complex XSLT Scripts," shows some useful patterns for building flexible and reusable stylesheets. One way to speed up integration is to efficiently access and manage data in XML format. However, no matter how easy it is to get access to data and to move it between applications, there is always the business logic to worry about. While it remains buried inside custom applications, the cost of implementing changes to integration projects will be high. The solution is to externalize as much of the business logic as possible through workflow specifications and business rules. The move toward standardizing Web services choreography is a step in the right direction as Suhayl Masud shows in his article, "Building a Real-World Web Service," which applies BPEL to RosettaNet processes.

This issue takes a balanced look at the progress we're making and the direction we need to go to solve integration problems, along with some of the obstacles we're bound to encounter along the way.

AUTHOR BIO

Simeon Simeonov is a Boston-based principal at Polaris Venture Partners with a focus on opportunities in information technology. A founding member and chief architect at Allaire, Sim led the development of ColdFusion. He was activley involved with standards dvelopment through W3C, OASIS, and JCP.

TALKTOSIM@POLARISVENTURES.COM

SYS-CON MEDIA
135 Chestnut Ridge Rd., Montvale, NJ 07645
TELEPHONE: 201 802-3000 **FAX:** 201 782-9637

PRESIDENT and CEO
Fuat A. Kircaali fuat@sys-con.com

BUSINESS DEVELOPMENT
VP, Business Development
Grisha Davida grisha@sys-con.com

COO/CFO
Mark Harabedian mark@sys-con.com

ADVERTISING
Senior VP, Sales & Marketing
Carmen Gonzalez carmen@sys-con.com
VP, Sales & Marketing
Miles Silverman miles@sys-con.com
Advertising Director
Robyn Forma robyn@sys-con.com
Advertising Account Manager
Megan Ring-Mussa megan@sys-con.com
Associate Sales Managers
Carrie Gebert carrieg@sys-con.com
Kristin Kuhnle kristin@sys-con.com
Alisa Catalano alisa@sys-con.com

SYS-CON EVENTS
President
Grisha Davida grisha@sys-con.com
Conference Manager
Michael Lynch mike@sys-con.com
Regional Sales Managers, Exhibits
Michael Pesick michael@sys-con.com
Richard Anderson richarda@sys-con.com

CUSTOMER RELATIONS
Customer Service Representative
Margie Downs margie@sys-con.com

JDJ STORE
Manager
Rachel McGouran rachel@sys-con.com

WEB SERVICES
VP, Information Systems
Robert Diamond robert@sys-con.com
Web Designers
Stephen Kilmurray stephen@sys-con.com
Christopher Croce chris@sys-con.com
Online Editor
Lin Goetz lin@sys-con.com

ACCOUNTING
Accounts Receivable
Kerri Von Achen kerri@sys-con.com
Financial Analyst
Joan LaRose joan@sys-con.com
Accounts Payable
Betty White betty@sys-con.com

SUBSCRIPTIONS
SUBSCRIBE@SYS-CON.COM
1 888 303-5282
For subscriptions and requests for bulk orders, please send your letters to Subscription Department
Cover Price: $6.99/issue
Domestic: $69.99/yr (12 issues)
Canada/Mexico: $89.99/yr
all other countries $99.99/yr
(U.S. Banks or Money Orders)
Back issues: $12 U.S. $15 all others

Can Software Cross the Standards Divide?

WRITTEN BY **ERIC NEWCOMER**

No standards movement in the history of the software industry has garnered as much attention or support as Web services. After the previous decade's failed attempts to reach unity, the industry has lauded the promise of more flexible, open, and interoperable software as a revolution and a breath of fresh air.

Corporations are applauding the possibility that the software industry is finally starting to standardize its offerings, thereby helping to eliminate the islands of information that exist within nearly every Global 2000 organization. It's perceived that Web services will go a long way toward eradicating the integration challenge that's the number-one cause of CIO headaches.

Despite many attempts, and some pockets of progress, the software industry remains essentially a craft business, relying on the individual knowledge and skills of highly trained, experienced developers to stitch together disparate operating systems, middleware systems, and development environments. The largest cost of IT projects is labor. The industry needs standards that allow rapid mass assembly of individual components into integrated business operations.

Many industries have gone through this process, for example, standard shipping containers in transportation, standardized parts in automobile and hard goods manufacturing, standardized components in PC manufacturing, and standardized media formats in CD and DVD entertainment systems. In each case, a tremendous shakeout occurred in which new companies gained market share by embracing the new standards and understanding their true business value.

Today we're at a crossroads. The core standards – SOAP, WSDL, and UDDI – have been universally adopted and implemented, but disagreements over next steps and industry fragmentation are emerging over the standardization of technologies at the upper layers of the stack. Recently, the industry has witnessed a controversy around orchestration, or choreography, with Oracle and Sun leading the WSCI (Web Service Choreography Interface) effort and Microsoft and IBM leading the BPEL (Business Process Execution Language) effort. We also have Oracle and Sun publishing WS-Reliability without involvement from IBM and Microsoft, citing intellectual property concerns. It's turning out to be much harder to gain agreement on significant next-level features, such as security, transactions, reliable messaging, management, and orchestration, than it was to gain widespread agreement on the core standards. Some software vendors base their entire businesses on these higher-level features and functions, while others fight strongly to differentiate themselves from the competition by delivering better implementations.

Standards are necessary to break the lock-in of proprietary systems. But they are most significant because they improve the speed of assembly, as they did in automobile and hard goods manufacturing. Or because they increase the efficiency and speed of operations, as standardized containers did for the shipping industry. And above all because they reduce the overall labor costs relative to the production and sale of the finished item, which in the case of software is the operational IT system or integration.

One issue preventing agreement is intellectual property rights. Microsoft and IBM appear to be holding out to maintain intellectual property rights on the specifications they create, while Oracle and Sun have gone on record in favor of unrestricted access to the specifications they create. Depending on how this issue is resolved, the software community will either be able to implement new Web services specifications for free, or be required to pay for the privilege.

It's a valid question: Do a specification's authors have the right to profit from its implementation by others, thereby recouping some of the value of their work? Another valid question arises when a specification defines technology that a company has previously implemented in a product and perhaps protected via patents. Should that company have the right to charge a license fee to permit other companies to implement the specification based upon it?

These questions get to the heart of some of the current disputes slowing progress on specifications such as choreography and reliability. (Choreography is significant because it represents the all-important reusability value of Web services by stitching individual services together into composite applications, by business analysts rather than skilled programmers; reliability is important because businesses need to know whether or not their requests were received.)

Rather than view Web services as an open, new, and large market opportunity in which everyone can compete, established vendors naturally try to maintain their positions of dominance by controlling the extent of standardization.

The level of standardization applies to features within a particular technology area, such as security, transactions, or management. If the standard specification doesn't include a sufficient level of features and functionality, proprietary technologies may still be required to create

useful applications or technologies.

The extent of standardization applies to general technology areas viewed as appropriate for standardization. To use the automobile analogy, it's appropriate to standardize parts that need to be fitted together during mass assembly, but it's not appropriate to standardize parts related to the human interface – there is no standard size for a steering wheel, for example. In Web services, some might argue that choreography and reliability aren't appropriate areas for standardization, and thus remain a point of competition instead.

One could speculate further, given a broad historical view of how industries reach a sufficient level of standardization to reduce costs and improve efficiencies, as to whose interests are better served by delaying progress in standardization, or by confining standards to particular levels or areas. Services companies naturally do not want it to be too easy to integrate disparate systems, for example, as they have based their business model on the ongoing demand for high-priced labor. Operating system companies naturally do not want all areas of the operating system to be standardized, as their business model relies on the high margins involved in selling proprietary software products.

The software industry must now choose its path, and the chosen road will impact end users and developers differently. One road leads to a truly standardized world where corporations reap the economic benefits of Web services through the lower prices that open competition usually brings, drastically reducing IT costs. The second road leads to the world of proprietary systems, yielding massive vendor service and maintenance revenues, and killing end-user flexibility and future return on investment.

There are several reasons why the industry is in danger of veering down the wrong road. First, the business environment changes after standardization, and the software industry is having trouble coping with this reality.

Another reason is the impatience of vendors to profit on their Web services investments – the Web services market is not taking off as fast as everyone initially thought it would. It's ironic that some of the same people who promote Web services as the next revolution are impatient as they discover that the technology will become mainstream, which by definition means it will take time to become adopted by conservative IT shops.

The biggest reason for the industry's tendency to choose the wrong path is that the industry is having difficulty drawing the line between what is appropriate for standardization and what should remain as a basis for competition. At the basic level, such as agreeing on something fundamental like TCP, commercial interests don't override the universal needs of the computing industry. Thus the core specifications (SOAP, WSDL, and UDDI) were fairly quickly agreed upon and implemented. We're now beyond that level and are entering an arena where commercial interests are much stronger.

To accomplish software industry standardization, vendors have to shift focus from selling proprietary products with a "standards compliant" label on them to focus instead on cooperating to create a larger market based on truly standard products. In the CD or DVD industry, anyone's disc works with anyone's player because the electronics companies and entertainment content vendors were able to agree upon a common format for the entire product. We are still somewhat far away from the day when a program written on any computer will work on any other as seamlessly as a DVD will play in any hardware system supporting that format. Software vendors, unfortunately, continue to focus on differentiation and long-term relationships based on proprietary features rather than on creating a broad new market.

What's needed is agreement that the potential Web services market is bigger than all current software markets combined. Some vendors prefer to draw the standardization line where it currently sits, with SOAP, WSDL, and UDDI. But customers correctly say that this level of standardization isn't enough to realize the promise of Web services technology.

Many software vendors quite naturally portray Web services as merely another feature of existing products, since they often have millions of dollars of investments at stake in those products. However, Web services more correctly represent an XML layer on top of any and all existing software systems, programming languages, and packaged applications, and that layer has unique requirements that need to be recognized and standardized.

Market leaders often introduce new specifications under the guise of better industry-wide standardization, when in actuality they are more concerned about mapping the specifications to their own product sets. They try to agree on the minimum possible to call something a standard while still requiring customers to use proprietary features. The goal is to blur the line between their products and what is accepted as the standard, so that they can sell their software as "standards based." In essence, the customer is again buying proprietary software that facilitates the same level of vendor lock-in as the prestandard offerings, if not more. Not only does this not solve the integration challenge, it actually exacerbates the problem.

Established vendors naturally try to subjugate the greater good and customer benefit for their proprietary self-interests. They're afraid of industry changes that invalidate their investments in current products. If Web services standards are comprehensively adopted, not only at the basic level but also at the extended levels, it will change the economics of the software industry and make their investments in current products unsustainable.

Because Web services represents a revolution in the way software and systems are designed and interact, it threatens the existing franchises of major vendors in markets that include application servers, application development, and application integration. The choice for these vendors is to start over and risk losing market share or to try to control the revolution. Not surprisingly, these companies are choosing the latter. It took television years to gain widespread adoption, in part because of how strongly radio manufacturers opposed it.

Standardization of parts for the mass production of automobiles did not take off until Henry Ford insisted upon it by enforcing strict purchasing policies with his suppliers. It's very possible that the Web services standards effort will continue to flounder until businesses truly start using Web services to talk with each other as often and frequently as they use the fax and phone, and they start to put real purchasing pressure on the vendors.

The next year will be important and telling in the development of Web services. End users and independent software vendors need to demand that standards remain neutral and not map to a small handful of vendor specifications.

The promise and goal of Web services is to unlock the door to past IT investments, change the proprietary nature of new software offerings, and radically change the economics of the software industry. Without question, those ideals face tough obstacles over the next twelve months.

AUTHOR BIO

Eric Newcomer is CTO of IONA Technologies. He is a prominent member of a number of standards organizations and a staunch proponent of standards neutrality.

ERIC.NEWCOMER@IONA.COM

PolarLake

www.polarlake.com

Reader Feedback

SOAP vs REST

REST is a fundamental alternative to the whole SOAP/WSDL/UDDI stack. The idea with REST is to establish a convention for a URI that is used for discovery. Then if you want to know what services are published by acme.com you ask for http://acme.com/wesvc_catalog or some such. This serves back an XML document that enumerates the available services, each with a URI to ping for more details. Then you ask for http://acme.com/websvc_catalog/order_wockets or some such. That returns an XML document specifying the exact interface to the "order wockets" service.

The point is that each service and each level of discovery/description has its own URI, which means that the requests can be passed through by the transport layers without having to open and parse the payloads. The problem with SOAP as it's done is that most schemes envision the message payloads having to be repeatedly opened and parsed. Imagine if IP packets could not be routed on the basis of the header alone, and you will begin to understand what the problem is.

Frank Wilhoit

XML on MSOffice, a Really Big Step

MS has taken a very big step in changing its Word format to XML. Word is a markup language that is closed; XML is very open! Using an open format means more competitors in the arena. It also means that electronic integration of many products is just around the corner. Yes, it's true that there is an open project called OpenOffice, but we must consider that MS has a really big market, and those numbers speak for themselves. OpenOffice will stay strong, but MS is still winning for now.

Henry Villca

Letters may be edited for grammar and clarity as well as length. Please e-mail any comments to Hitesh Seth (hitesh@sys-con.com).

XML and Integration

WRITTEN BY **HITESH SETH**

Better known in the *i*-technology world as enterprise application integration (EAI), B2B integration, or middleware, integration involves connecting internal systems with external business partners, customers, and suppliers. Integrating systems running on heterogeneous platforms, typically developed in different programming environments and managed by different groups (or different companies), is quite a complex task.

And guess what technology has been used to solve this complex problem? XML. Truly, the integration services space is probably the largest application area of XML. XML provides the world of integration with an open and extensible standard to define and implement loosely coupled business documents and processes.

To appreciate the value that XML brings to integration, I decided to talk with leading integration software providers who have used XML in various incarnations (a.k.a. markups) within their products and have extensively leveraged the benefits of XML.

Self-Describing Data Is the Key

Not surprisingly, there was a consensus among competitive vendors about what has been the biggest benefit of using XML in their core technologies. XML's success as the open-standard mechanism for self-describing data is most significant. The ability to carry metadata with data is particularly important in integration that involves sharing data within a heterogeneous environment where there isn't even a basic common element, such as data types. This "data-typing" information (XML Schemas) allows data to be efficiently routed and processed intelligently. This is a radically different approach to solving the integration problem – before XML, integration was all about connecting individual systems to each other, in most cases using a proprietary mechanism.

Toward Loosely Coupled Architectures

Another fundamental reason XML is important, which builds on the fact that it is self-describing, is that by using XML we're able to facilitate a document-centric, loosely coupled model for integration instead of the more tedious RPC/API-centric model. This model resembles in many ways how enterprises have functioned for years using a paper-based (document) model. This approach makes integration a lot easier to sell to businesses that conceptually agree with it. A natural extension of this loosely coupled architecture is really what the whole Web services phenomenon is about.

Web Services Orchestration – Silver Lining?

In a number of ways the emerging (or should I say already emerged?) Web services space has close ties to the world of integration. The core standards for Web services – XML, SOAP, and WSDL – are close to being "officially stamped," and lately there has been a lot of activity around establishing a standard mechanism for tying Web services together (a.k.a. orchestration of business processes). This is increasingly thought of as the missing link in the current integration and Web services technology infrastructures. Of course, this isn't an easy problem to solve; in addition, it's not an easy standard to get industry consensus on. Of course this is natural because there's inherent complexity as we move up the stack.

"XML has provided a means to codify and standardize interfaces that is driving a new level of software interoperability"

Isn't XML Verbose?

What about XML's being too verbose, which can in some cases lead to a performance hit? This is definitely a concern in some scenarios. Most vendors accept this fact; some consider it a serious problem and even suggest the need for a standard for compact/binary representation of XML. However, most don't consider a revolutionary change necessary. It's mainly evolutionary, and most of the vendors feel that, in line with Moore's law, growing processing power and improvements in parsers will solve this problem over time.

What About EDI?

It's true that EDI documents are still being utilized, and this will continue. The core benefit of EDI, however, is not the format but the fact that over the 20+ years during which EDI has evolved and been implemented by thousands of companies, it has established ubiquity

in the business world. Much of the benefits of EDI lie in the fact that it's been used as the de facto format by numerous applications that have "built-in" support for the standard. Compared to EDI with its 20+ years of experience, XML is still a young child at 5 years old. XML does have technology advantages over EDI, but most of the integration players see EDI and XML as coexisting technologies, where EDI will continue to be used within established interfaces (applications/business partners) and XML will be gradually injected as the downstream integration mechanism. The presence of EDI systems was compared to the existence of the mainframe. Mainframe-based systems aren't going away, and neither is EDI. What started as coexistence will, however, gradually move toward replacement, as we have consensus in required transaction management, security, and quality of service infrastructure and standards. In addition, XML/Web services solutions will gradually emerge as much more financially optimal.

"Wide adoption of XML-based standards such as Web services could eliminate historical barriers to interoperability and drastically reduce the price of integration"

Common Business Markup – Not Yet a Reality

In XML's early years in the world of integration, we saw a number of industry initiatives that attempted to standardize common business documents within a particular domain. Some of these initiatives have reaped benefits, while others weren't able to harvest enough interest. However, apart from these industry-specific vocabularies, we've seen initiatives in action that attempted to create a cross-domain standard for business documents and in some cases even business processes – similar in a sense to what EDI has accomplished in the supply-chain world. In this area, most of the vendors feel that we're probably better off working on industry-specific standardization initiatives such as RosettaNet, CIDX, HIPAA, HL7, OFX, SWIFT, etc. Consensus on a standard for horizontally applied common business documents and processes is quite far away.

Vittorio Viarengo
Director of Product Management, BEA Systems, Inc.

XML is the key technology that is enabling a whole new way of integrating applications and systems: loosely coupled integration. XML, and related standards such as XSchema, XSLT, and XQuery are bringing the cost of integration down by providing enterprises with a set of standards-based technologies to describe, query, and transform data. This makes it possible to tackle integration challenges that before XML were either too costly (proprietary technology, lack of skills) or not feasible (B2B integration over the Internet). Moreover XML's extensibility makes it easier to accommodate changes driven by ever-evolving business requirements and to integrate any type of data sources from structured (databases) to semi-structured data (content).

Tommy Joseph
CTO, TIBCO Software Inc.

XML and related standards provide a standards-based expression for several of the concepts that lie at the core of the integration problem. For example, TIBCO pioneered the use of self-describing data for application integration. This concept lies at the heart of the XML standard. Other examples include the concepts of data transformation and service-oriented architectures. These concepts are expressed through the XSLT and WSDL standards, respectively. As these standards have gained acceptance, TIBCO has made them increasingly central to its strategy.…The advantage of standardization is that it makes it easier for more and more services to be plugged into TIBCO's integration platform.… On the other hand, integration is by definition a heterogeneous problem. The real world is composed of heterogeneous systems. So TIBCO's platform also includes support for other standards like J2EE, Microsoft's .NET and COM, CORBA, and a wide variety of legacy and proprietary technologies.

Dave Wascha
Lead Product Manager, E-Business Servers, Microsoft Corporation

The key value of XML is that its self-describing nature helps simplify integration. It might not be the best technically, but it has a lot of industry consensus. XML is still a relatively young technology. What we have seen so far is only 2-3 years of credible usage. We have a long way to go and we still have much to realize in tangible benefits.… With the core standards around XML/Web services – SOAP/WSDL/UDDI being broadly agreed on – and the next set layer of standards in the stack close to being finished, we are about to hit the acceleration point of adoption. New technologies and products will be key in making the barrier to entry low. Even mom-and-pop shops, which form 80% of the global supply chain, need these technologies to be efficient, and they don't have much money to spend. We need real products that solve real business problems in a cost-effective way.

John Magee
Vice President, Oracle9*i* Middleware and Tools, Oracle Corporation

XML has become the key foundational technology for integration. Its powerful descriptive capabilities, combined with its relative simplicity, make it an ideal tool for addressing a wide variety of integration challenges. From data and application integration to distributed computing and Web services, XML has provided a means to codify and standardize interfaces that is driving a new level of software interoper-

ability. As core standards such as Schema, XSLT, and XPath continue to solidify, and as emerging standards for Web services security, orchestration, reliability, and adapter-based integration architectures mature, XML-based middleware will provide a platform for automating business processes that span system and organizational boundaries.

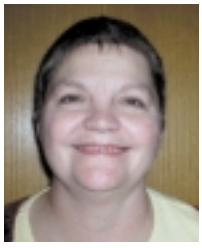
Dale Skeen,
CTO,
Vitria Technology, Inc.

Adopting XML inside and outside the firewall makes some of the lower-level integration problems typically associated with data formats go away. This adoption is forcing companies to acknowledge and confront the harder problems in integration: What are my business semantics? How do I orchestrate with my business partners?…Emerging XML technologies, particularly around Web services, are definitely becoming pervasive, despite the existence of certain technical issues, especially around security and reliability. The real problem going forward lies not with these technical issues, which have known solutions, but with getting consensus and political buy-in for a single solution to each issue.

Eric Newcomer
CTO,
IONA Technologies

Wide adoption of XML-based standards such as Web services could eliminate historical barriers to interoperability and drastically reduce the price of integration – and speed time-to-market through rapid application assembly. Because the standards process is currently controlled by the software manufacturers and not the customers, however, it's very possible that progress toward reaching this goal will be slow rather than quick, since the manufacturers naturally protect established practices and business models. Industries tend to reap the benefits of standardization only when the consumers insist upon them, and this has yet to be the case for Web services. An open, end user–driven standards process will soon become a necessity to ensure that the industry progresses rapidly to produce the standards-based integration solutions that companies are demanding.

Rachel Helm
Director, Product Management,
WebSphere Business Integration,
IBM Corporation



Many IBM customers have chosen to revitalize their EDI solutions rather then completely replace them. Why? Because they've got tremendous investment in the semantic content of those EDI transaction formats they defined with their partners. Why abandon those partner agreement formats while they are still applicable? The VANs, on the other hand, are increasingly not cost-effective. So our customers move their existing EDI transactions over to an EDI/INT platform in order to have the best of both worlds: leveraging of still-relevant transaction document formats along with the cheap, pervasive technology of the Internet. What technology do they use as the data format bridge? You guessed it – XML.

Jim Green
CTO,
webMethods, Inc.

In the pre-1995 era in middleware, before EAI became a topic, the middleware space consisted of middleware products that view data as a BLOB, and we didn't have any ability to carry structure with the data. Both the sending and receiving applications knew about the proprietary structure, but there wasn't any easy way for a third-party application to know what was going on. XML truly emerged as an unambiguous standard to represent structure with the data. Including structure with the data provides a schema representation to the middleware that is fundamental to the technology and its ability to route data based on the type. XML was required to enable today's integration technologies.

David Litwack
Senior VP, Web Application Development Products,
Novell, Inc.

XML is key to developing today's services-oriented applications. These are composite applications that pull information from a variety of existing and new data sources and transform them into business uses for which they were never originally intended. Going forward, an ever-increasing percentage of information that flows across networks will be XML-based. To date, the industry has been largely focused on the creation or production of Web services, but in the next several months I think we'll see increased focus on the consumption of services. For example, emerging standards like XForms will enable data binding of XML to presentation, thereby providing end-to-end production and delivery of services. This is where it starts to get interesting, as these new technologies empower a broader range of developers to quickly build services-oriented applications.

Conclusion

XML has truly emerged as one of the most significant technologies changing the landscape of technology itself. What was conceived as simple markup language has evolved as a ubiquitous data model for almost anything. XML is still, however, five years young. We have yet to realize the benefits of XML and its applications, particularly Web services. With the core infrastructure around XML, Schemas, and Web services laid out, XML adoption is about to take off.

AUTHOR BIO

Hitesh Seth, editor-in-chief of XML-Journal *and XML Track chair for Web Services Edge, is the chief technology officer of ikigo, Inc., a provider of Business Activity Monitoring solutions.*

HITESH@SYS-CON.COM

Premiering
June 2003
at
JavaOne

www.sys-con.com

Millions of Linux Users One Magazine

Linux Business and Technology

There is no escaping the penetration of Linux into the corporate world. Traditional models are being turned on their head as the open-for-everyone Linux bandwagon rolls forward.

Linux is an operating system that is traditionally held in the highest esteem by the hardcore or geek developers of the world. With its roots firmly seeded in the open-source model, Linux is very much born from the "if it's broke, then fix it yourself" attitude.

Major corporations including IBM, Oracle, Sun, and Dell have all committed significant resources and money to ensure their strategy for the future involves Linux. Linux has arrived at the boardroom.

Yet until now, no title has existed that explicitly addresses this new hunger for information from the corporate arena. *Linux Business & Technology* is aimed squarely at providing this group with the knowledge and background that will allow them to make decisions to utilize the Linux operating system.

Look for all the strategic information required to better inform the community on how powerful an alternative Linux can be. *Linux Business & Technology* will not feature low-level code snippets but will focus instead on the higher logistical level, providing advice on hardware, to software, through to the recruiting of trained personnel required to successfully deploy a Linux-based solution. Each month will see a different focus, allowing a detailed analysis of all the components that make up the greater Linux landscape.

Regular features will include:

- *Advice on Linux Infrastructure*
- *Detailed Software Reviews*
- *Migration Advice*
- *Hardware Advice*
- *CEO Guest Editorials*
- *Recruiting/Certification Advice*
- *Latest News That Matters*
- *Case Studies*

SAVE 30% OFF!

REGULAR ANNUAL COVER PRICE $71.76

YOU PAY ONLY $49.99

12 ISSUES/YR

**OFFER SUBJECT TO CHANGE WITHOUT NOTICE*

SUBSCRIBE TODAY!

WWW.SYS-CON.COM

OR CALL

1-888-303-5282

LINUX EDGE conference&expo

June 3–5.....LONDON
June 24–26.....BERLIN
September.....HONG KONG
October.....CALIFORNIA

FOR ADVERTISING INFORMATION:

CALL 201 802.3020 OR VISIT WWW.SYS-CON.COM

SYS-CON MEDIA
The World's Leading Technology Publisher

Integration Is the Killer App

WRITTEN BY
SIMEON SIMEONOV

Putting the future in perspective

In 1975 Niklaus Wirth, the Swiss computer scientist who created the Pascal programming language, published a seminal book entitled *Algorithms + Data Structures = Programs*. If Wirth had written about business applications, *Computing + Storage = Applications* would have been a better title. Of course, in 1975 there weren't that many business applications. Most of them ran limited back-office functions on mainframes. PCs weren't on the map. The killer app for PCs – the first spreadsheet – wouldn't be created until 1981.

More than a quarter of a century later, things have changed. Most notably, applications are connected. TCP/IP and Ethernet provide a base set of networking standards. Distributed computing standards, XML and Web services being the latest ones, enable cross-application communication. A modern rendition of Wirth's statement could be *Computing + Storage + Connectivity + Open Standards = Applications*. In a world of connected applications, integration is the killer app. It is the key to growing the business value of applications. To really understand the significance of this claim, it helps to put the future in perspective by looking at the broader trends in computing, storage, connectivity, and open standards.

Trend Watching

Moore's Law – named after Gordon Moore, cofounder of Intel – has defined computing trends since 1965. Since its last revision in 1975, it states that the number of transistors per square inch on integrated circuits doubles every 18 months. Moore's Law is expected to hold for at least a decade longer. Processors, memory, and other circuitry will get both faster and smaller. Computers will crunch through more data. Mobile devices will have significant independent computing capabilities, enough to run basic office applications. Embedded microprocessors, already present in everything from smart cards to household appliances, will start appearing in ever more places. And, just as Moore's Law is starting to give way, quantum computing may be within reach.

Storage also exhibits powerful trends. Storage density has increased by nearly 50% per year. Total storage capacity has been increasing much faster – more than doubling every year. Costs are falling rapidly. Disk storage is at about the cost of tape storage – around $1/Gb. You can buy a terabyte drive for less than $1,000. At that rate, it will soon be possible to keep all data online all the time. Portable devices are also increasing their storage capabilities dramatically. CompactFlash cards of 1Gb are readily available. In the same form factor, Hitachi Microdrives provide several gigabytes of storage. For smart phones there are MultiMediaCards. With comparable capacities, they are smaller, more secure, and more efficient. In the future, we have new types of optical memory, for example, holographic, to look forward to.

Connectivity hasn't exhibited the predictable trends of computing and storage. Instead, it has exploded. The Internet boom was marked by massive investment in capacity. Dark fiber is still waiting to be used while telcos are going bankrupt to restructure their expansion debt. Most businesses and many households have broadband connectivity. Broadband capabilities are increasing. New technologies allow gigabit traffic through existing cable lines. Still, the progress in wired connectivity pales in comparison to the great strides in wireless broadband.

Businesses and homes are embracing untethered computing at increasing rates. The suite of 802.11 standards already offers secure bandwidth up to 56Mbps. 3G may be forever behind schedule, but wireless providers are deploying 2.5G and 802.11 hotspots are popping up in many areas. Two-way satellite offers decent connectivity even in remote locations. The FCC recently unlicensed spectrum around 60Ghz, which allows point-to-point wireless connectivity at gigabit speeds. Mesh networking eliminates the signal strength issues that surround point-multipoint approaches (such as 3G and satellite). At the other end of the wireless spectrum, connectivity is about lowering costs. Very low-cost self-organizing networks seamlessly connect devices such as industrial sensors without

wires and without configuration. Further down on the cost scale (at mere cents) are RFID tags that will allow, for example, FedEx packages to be tracked automatically as they enter and leave a distribution center.

Computing, storage, and connectivity are meaningless if they cannot be used together to build and connect applications. That's why we need standards. TCP/IP and Ethernet provide the base of networking standards on top of which we run naming services through DNS and DHCP, the Web through HTTP, e-mail through POP/IMAP, and many more. For application development the industry is converging to two runtime platforms: Java/J2EE and Microsoft's .NET. However, the biggest strides on the standards front have recently been made with respect to integrating applications. XML is steadily becoming the workhorse of data definition and representation. Soon, we won't be hearing much about XML. This will be a good thing, the final indicator that XML has found its proper place in the stack of commonly used technologies. Nobody talks about ASCII anymore; it's taken for granted. The same will happen with XML. Web services have some way to go to get there; there is still too much overpromising and underdelivering.

Despite the hype, one thing remains true: Web services are our current best hope for an open, flexible, and comprehensive set of standards covering distributed computing and application integration. First, all the major platform vendors are behind this movement. Second, the standards are designed in a way that supports the distributed evolution of related specifications by different vendors and different standards organizations. This industry process, although it is not a pure "one company, one vote" democracy, is a significant improvement compared to the traditional extremes of authoritarianism (DCOM) or design-by-consensus (CORBA).

There you have it. The future will offer much computing power and storage spread along an ever-increasing number of devices. The numbers of units of every new generation of computing devices have dwarfed previous numbers: from mainframes to PCs to handhelds and phones, and now all things with capable microprocessors. Plenty of connectivity and a solid base of standards will enable many distributed applications. New grid architectures will provide abstractions that focus on what services are needed as opposed to where they are located and how they are accessed.

If we look at the trends, the future is bright. All this innovation embodied in connected distributed applications running everywhere – wow! What's wrong with this picture? There is a key ingredient missing – integration. The future can bring great value, but integration is the key to unlocking the potential of this future world.

Metcalfe's Law Redux

To understand the role of integration, it helps examine one of the pillars of the Internet, the network effect argument. It goes something like this: all computers connected to the Internet are going to leverage one another's capabilities and create enormous value. The proponents of this argument found support in a digital "law" proposed by Bob Metcalfe, inventor of the Ethernet and founder of 3Com, circa 1980. Most people believe that Metcalfe's Law states that "the value of a network is equal to square of the number of its users." Value is expressed as a monetary equivalent – some multiple of U.S. dollars, for example. By that argument, when the Internet had 100M users, its value was equal to some constant multiple of 10^{16}. Surely, the multiple cannot be too big since that amount far exceeds the overall output of the world economy.

Bob Metcalfe is a partner in my firm and he gave me a copy of the original slide describing Metcalfe's Law (see Figure 1). In its original form, the statement was, "The systemic value of compatibly communicating devices grows as the square of their number." The law is based on the mathematical truth that there are $N(N-1)$ possible point-to-point connections between N computers on a network. Value is derived through "compatible communication," a meaningful information exchange between the machines. The law assumes that all connections carry equal value. While not true in the real world, this simplifying assumption does not change the fundamental insight of Metcalfe's Law. Metcalfe's Law defines a theoretical maximum for the value of a network. Because value is derived through the leverage of meaningful connections between machines, in a network where only a few machines are regularly exchanging information, the actual network value will be significantly lower than the potential allowed by Metcalfe's Law.

The main difference between what Bob Metcalfe said and how it is remembered has to do with how the size of the network is measured – devices versus users. This turns out to be a crucial difference. Machines can easily scale the number of connections or conversations they have with other machines. They are good at processing a lot of information. Humans cannot do that. The brain is not designed for it. For example, while my computer at home can access hundreds of Web sites in just a matter of minutes, I, its user, could neither handle the fast context switching nor process the amount of information presented. Humans get overloaded with information fairly quickly. In the digital domain, people use server-based applications

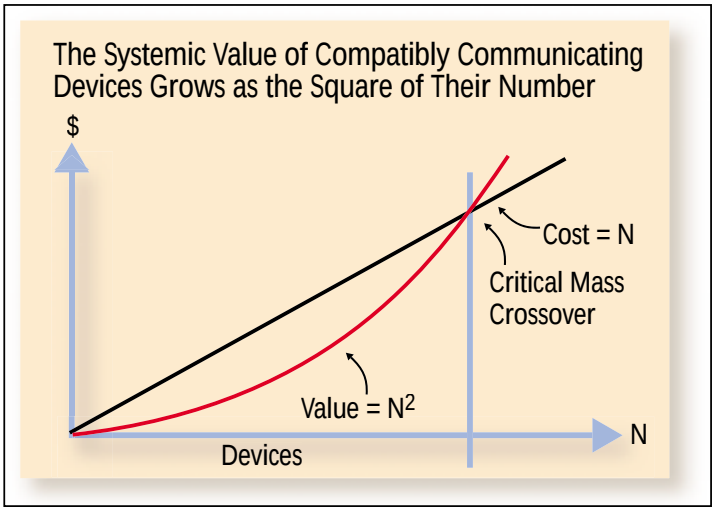


Figure 1 • Metcalfe's Law

that aggregate and integrate information. Search engines and portals on the Web are good examples as are integrated information systems in corporations. As the number of sources and amounts of information increase, people need more help using this information. As the size of networks goes up, servers and the connections between them – not the number of users of the network – offer the most potential for exponentially increasing the value of networks. And the key to that potential is integration.

Analyzing Network Value

A simple network model demonstrates the value-creating potential of integrated servers. Assume that in a network of N machines, s denotes the percentage of servers each running

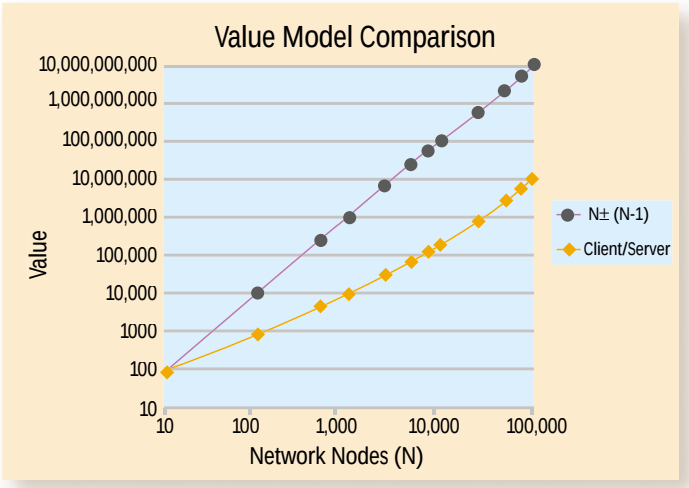


Figure 2 • Value model comparison for client/server network

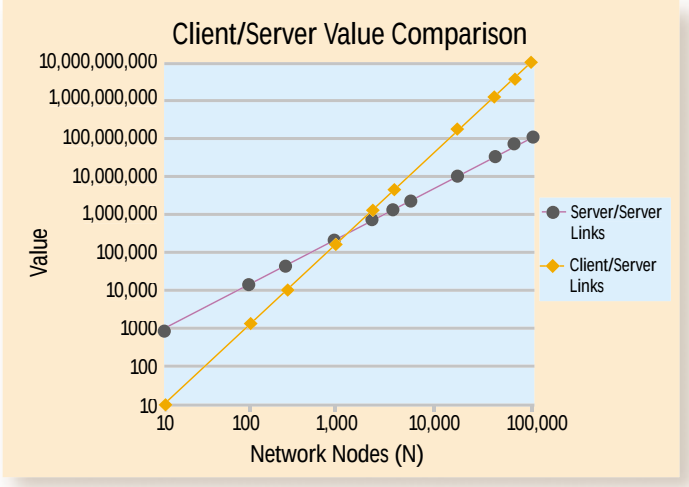


Figure 3 • Client vs server connections

one application. The total number of servers is sN. On average, these servers are integrated with a percentage of the rest of the servers, i. The total number of server-to-server connections is $i(sN)2$. The clients for these servers are desktop computers. Clients cannot communicate directly between themselves. They can only communicate with servers. On average, each client communicates with k servers. The total number of client-to-server connections is $k(1\text{-}s)N$. Combined, the client-to-server and server-to-server connections yield the total number of connections for the network.

For example, let's choose a network with 90% clients and 10% servers. The integration ratio between servers is 10% and clients can communicate with 10 servers on average. Figure 2 shows how the value of this network compares to the ideal value allowed by Metcalfe's Law for networks between 10 and 100,000 nodes. Both axes use logarithmic scale to help with comparisons for lower values of N. As the network grows, its value is mostly generated by connections between servers – $i(sN)2$ – because the connections scale proportionately with the square of the number of servers. Figure 3 demonstrates that after about 10,000 nodes server connections contribute more value than client connections whose value only grows linearly with the number of clients.

This insight explains the difference between large and small networks. For small networks, the value is dominated by clients. However, in very large corporate networks or networks the size of the Internet, the mere addition of more users does not increase the value of the network more than linearly. On the other hand, the addition of meaningful applications that (1) leverage the value of many other applications and (2) can be accessed by all users has significantly more impact. This makes sense when applied to the Web. Do you personally get more value out of the few thousand servers that run in the Google data centers or a million broadband-enabled households in Korea?

Metcalfe's Law holds for the example network because server connections grow as the square of the number of network nodes. However, the value of the network is far lower than the potential allowed by Metcalfe's Law. As N grows, the ratio between actual value and potential value is equal to $is2$. For the example network this equals 0.1% or 1/1,000 of the true value potential (see Figure 4). Value grows as the square of the percentage of servers on the network. Also, the integration ratio, i, is directly related to the value-generating potential of networks. The more servers are integrated together the more value will be generated in the network.

The Killer App

This simple network model may not apply to networks the size of the Internet. If N is 100M, then 10M servers will have to each communicate with 1M servers. Network boundaries, computational limits, and application as well as integration complexity make this impossible in the general case. If there is a limit to the number of meaningful connections that any one server can make, then Metcalfe's Law will no longer apply to the network. Value will still increase but at a linear, not quadratic, rate.

Even in that case, however, integration remains the best hope for increasing the value of the network beyond just increasing its size. Integration can happen on a larger scale than any user-centric information processing. It is about meaningful connections between network nodes and thus leverages the core insight of Metcalfe's Law. That is why integration is the killer app (in a meta-sense) in a world with an ever-increasing number of applications running on ever larger networks.

The need for better integrated applications is huge. Here are just a few examples from the public Internet and corporate networks:

- Have you ever forgotten some of your passwords? If single sign-on is broadly deployed you will not have to remember so many passwords.
- Have you ever chosen not to buy from a Web site offering you the best terms because you didn't have an account set up with them and you didn't want to spend the time registering? If eWallets work broadly on the Web this will not be a problem.
- E-mail is the lifeblood of companies, and much relevant business information is sent and received through e-mail, often outside the context of the enterprise systems. A whole industry has sprung to address this lack of integration across multiple domains: sales automation, collaboration, resource planning, and so on.
- One large financial institution takes more than a week to process a change of address. The institution thinks that they can make up to 10% more per customer if only they could process the change in less than 24 hours. They have been

trying for years, without success, to integrate several applications to do this.

- Forrester Research estimates that more than 50% of Fortune 1000 enterprises have more than 50 legacy or packaged applications that need to be integrated – in spite of the functional and business needs that require them to be. Given the simple example of the financial institution above, it would seem that the opportunity cost of not integrating these applications is huge.
- *CIO* magazine's tech poll recently showed that 87.5% of surveyed companies have an IT application backlog, weighed heavily toward integrating existing systems as opposed to buying new systems.

End users as well as IT and business executives want more integration and information leverage between the applications they use every day. More integration is needed everywhere. The integration market is huge. IDC estimates that in 2002, the size of the application integration software market in the U.S. was approaching $5B. At the same time, the U.S. systems integration market was about $38B. Most of that is pure labor cost – as opposed to hardware and software cost – and much of it is spent on integrating existing systems as opposed to building new systems. It's difficult to estimate how much money is spent in-house on application integration. It's certainly likely to be a significant portion of large corporations' IT budgets. Two things become clear from the analysis. First, the total market size ends up being in the tens of billions for the U.S. alone. Second, labor accounts for the majority of that amount.

The Service Constraint

To harness the value potential of applications, we need a lot more integration. The only way to achieve this is to significantly lower the cost of integration across the board. There is a problem, however. The low product-to-service ratio in the integration space puts a constraint on both the rate of growth of the market and the cost of integration projects. Screen scraping, data mapping, and building bridges between incompatible APIs are not scalable activities. They require trained personnel and a lot of time. These activities simply cannot keep pace with the trends described in the beginning of this article. Also, given the high service component of integration projects, even big decreases in the costs of integration products cannot have a significant impact on the total cost of integration projects.

When technology really picks up its pace, IT services cannot keep up. Think back to the days of the Internet explosion. It became nearly impossible to hire qualified people in 1998–1999, either full time or as consultants. Salaries were increasing while projects were failing in the hands of unskilled programmers and business analysts. During a technology recession the pressure is released, but this temporary downward shift in demand cannot change the basic fact that skilled labor is scarce. Unless enterprises fundamentally change their dependence on integration services they will not be able to leverage the powerful trends in computing, storage, connectivity, and standardization. Nor will they be able to leverage the full value potential of integrated distributed applications. Companies will be stuck with integration backlogs that forever exceed their resources, both monetary and human.

It's important to understand that XML and Web services cannot significantly increase the product-to-service ratio by themselves. Standards in that area do help eliminate low-level tasks such as forming messages in proprietary binary protocols and programming data transformations in low-level languages. However, XML and Web services are just tools, not end goals in themselves. If two companies use different schemas to describe what a customer is and have different Web services APIs for manipulating customers in their CRM systems, then somebody has to spend time to define the data transformation between the data types and to develop a bridge between the applications. This is high-level work that requires trained professionals. Standards will never, ever bring homogeneity in the way enterprises view information and in the way software vendors build applications. Competitive pressures always demand differentiation, which leads to incompatibilities.

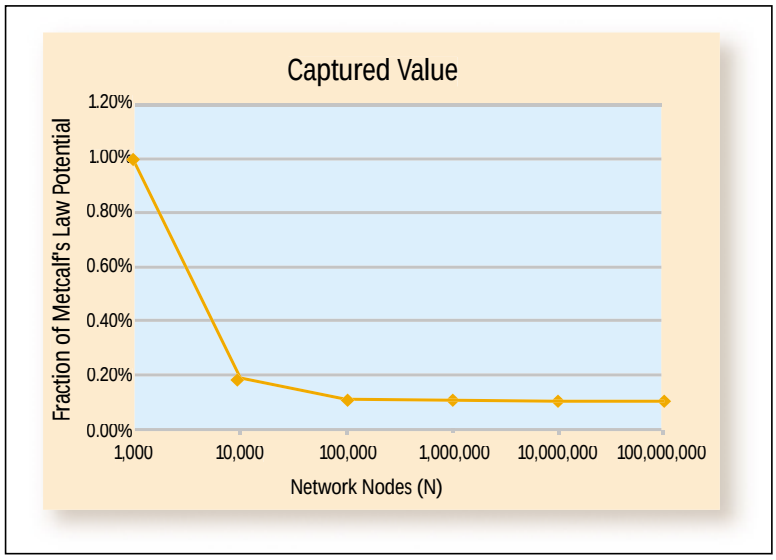


Figure 4 • Captured value convergence

The best approach is building better systems that significantly lower the planning, design, development, testing, and operating costs of integration. There is no single silver bullet. The efforts need to combine product development with standards work and partnerships between companies with best practices. People often think of integration as boring business. Nothing could be farther from the truth. It is a huge and fast-changing market with a desperate need for more innovation.

> "XML and Web services cannot significantly increase the product-to-service ratio by themselves"

In a follow-up article I will describe in more detail the areas that need attention and some of the promising technologies that can help. Also, I'll point out how some visionary companies, such as Microsoft, deeply understand that integration is the killer app of the future and are making long-term strategic moves to lower the cost of integration across the board.

Author Bio

Simeon Simeonov is a Boston-based principal at Polaris Venture Partners with a focus on opportunities in information technology. A founding member and chief architect at Allaire, Sim led the development of ColdFusion, one of the first Web application servers. Following the merger with Macromedia, as chief architect and VP of emerging technologies, he focused on Web services and other platform infrastructure for next-generation Internet applications. Sim was actively involved with standards development through W3C, OASIS, and JCP.

TALKTOSIM@POLARISVENTURES.COM

Building a Real-World Web Service

– Part 2

WRITTEN BY
SUHAYL MASUD

Choreography – making Web services dance

No, Web services isn't just hype. This series of articles will show you that Web services are real and can provide great benefits to organizations today. The series demonstrates how to build a real-world Web service by combining the features of Web services with components and knowledge of RosettaNet, an industry leader in e-business process standards.

I use WSDL (Web Service Description Language) and BPEL4WS (Business Process Execution Language for Web Services) to create Web services definitions for RosettaNet PIPs (Partner Interface Processes), and then use BPWS4J (Business Process Execution Language for Java Runtime) to implement these definitions. In this article, I will define the choreography for the real-world Web service by creating a BPEL4WS process that utilizes the Web service defined in Part 1. This article is all about choreographing the e-business dialogue, so put on your dancing shoes…the party's just begun.

Add a Little BPEL

In Part 1 (Vol. 4, issue 2), I showed you how to build a WSDL definition for a RosettaNet PIP. Along the way I discussed the true value of Web services, which is revealed in how they enable organizations to focus on services, and provide partners and customers access to these services through open and accessible public interfaces. I also talked about e-business dialogues and how to use the RosettaNet specification in Web services to conduct them. I ended the article by building a WSDL definition of the purchase order PIP from RosettaNet. In this installment, I'll take the real-world Web service definition further by adding BPEL4WS (written as BPEL from this point on) to the mix and defining an abstract process based on the WSDL definition. This abstract process will serve as a basis of the e-business dialogue definition that business partners will use to conduct e-business.

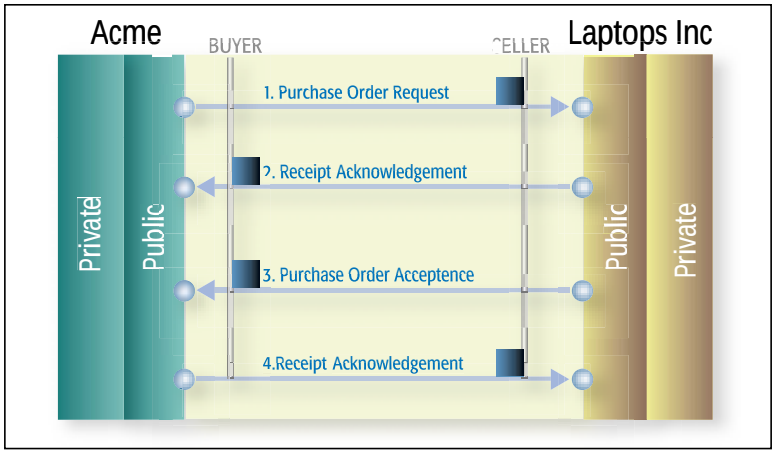


Figure 1 • Choreography makes Web services dance

I need to add choreography definitions to the e-business dialogue I'm building for the real-world Web service. The choreography will describe how processes and partners will interact with each other. These interactions exceed the limited functionality offered by the message exchange patterns available in the WSDL, which is why I'm using BPEL to pick up where WSDL left off.

The choreography information is central to the e-business dialogue, and there is a strong need for choreography languages to compose standard messages into meaningful, long-running e-business dialogues that can be used to conduct complete business scenarios. Choreography languages become more relevant as organizations begin to conduct e-business through their digital representations; it's simply not enough to have common message definitions (like those specified by EDI) and leave the choreography aspect of the dialogue to be handled over the phone or in face-to-face meetings. E-business dialogues are about interactions between digital representations of the business, and these representations need concrete choreography definitions (see Figure 1).

The Digital Representation of a Business

I have been talking about creating e-business dialogues between business partners. In my informal definition of the term, an e-business dialogue is two or more partners accomplishing a business goal (e.g., buying and selling goods) by conducting a dialogue between their digital representations. What are digital representations? In the crudest terms, a digital representation can be thought of as the computer systems of an organization that a business partner's application can connect to.

A better way to think about digital representation is to think about how an organization conducts business in the physical realm. Typically, the organization provides partners and customers access to its services through its physical representation (e.g., a storefront or an office building). Similarly, when this organization is conducting e-business in the digital realm (over the Web), it provides access to its services through its digital representation. This representation is more than access to databases, or connecting software systems. Organizations need an overall strategy for building service-based architectures, including digital representations reflective of their services.

As more organizations conduct e-business dialogues, they'll need to understand how the various e-business process and choreography languages work. Even at this early stage, there are several choreography languages to choose from, and the task can be daunting. To help in this cause, this article provides information about how BPEL, RosettaNet, and ebXML handle e-business dialogues and choreography.

Choreography Languages for E-Business Dialogues

The e-business dialogue is known by many names in the various choreography and business process languages; Table 1 relates these definitions.

To understand what role Web services, ebXML, and RosettaNet play in the e-business dialogue, see Figure 2, which originally appeared in "The Truth About Web Services," a report by Ted Schadler of Forrester Research (used with permission).

This section describes how the e-business dialogue and choreography are handled in more mature e-business languages like RosettaNet and ebXML. Currently, several languages deal with creating choreography using Web services (WSCI, WSCL, and BPEL); from this group I will explore BPEL.

RosettaNet

RosettaNet standardizes common e-business dialogues, capturing industry best practices for conducting business processes in the form of PIPs, which greatly reduce the setup time for starting a dialogue between business partners. RosettaNet specifications include the e-business dialogue specifications (the PIP), dictionaries that standardize usage of the terms used in messages, and an implementation framework that describes how to pack, unpack, and transport messages.

Here's how business partners conduct an e-business dialogue using RosettaNet: in order to conduct RosettaNet PIPs, both partners need to adhere to the RosettaNet implementation framework that describes how to pack and transport messages using open standards like HTTP and S-MIME. The e-business dialogue setup is simple – the partners agree on which RosettaNet e-business dialogue (PIP) to use, and they start conducting the dialogue over the Internet by exchanging secure XML-based messages. The PIP specifies a public process, the choreography of messages, definitions of the messages, definitions of process success and failure, and what to do when faced with exceptions.

EbXML

EbXML is built upon the experience and knowledge of EDI and RosettaNet, and in a sense, it combines the best of both worlds. EbXML standards include the BPSS (Business Process Specification Schema) to define e-business dialogues, a transport and routing specification, capability profiles that describe what e-business dialogues an organization can conduct, and a registry to store specifications and capability profiles. EbXML has received a boost from RosettaNet as well, as RosettaNet has expressed interest in using the ebXML BPSS and messaging services to define and execute PIPs.

To conduct a typical e-business dialogue with ebXML, business partners need to have implemented the ebXML infrastructure, and they should have registered a capability profile with the ebXML registry. This profile expresses the partners' ability to conduct an industry-standard e-business dialogue written in the ebXML BPSS. To conduct an e-business dialogue, the partners would find each other's capability profile on the registry and form an agreement about which dialogues they will conduct. These dialogues can be simple (like RosettaNet PIPs), or elaborate compositions of other dialogues. All messages are XML-based, packaged with S-MIME and a little bit of SOAP, and exchanged in a secure manner over public networks like the Web.

Web services choreography languages

The Web services choreography languages are built on top of the WSDL specifications. Reusing the operations and transport information, they focus on providing choreography by composing stateless Web services into state-aware e-business dialogues. Figure 3 relates RosettaNet, ebXML, and Web services choreography languages.

Choreography languages like WSCI, WSCL, and BPEL are all trying to do the same thing – make Web services more useful for business, even though what they do may be known by different names (e.g., orchestration, composition, choreography, flow). In the current languages, BPEL seems to have a lot

Standard or Specification	Defines e-Business Dialogue as
EDI (Electronic Data Interchange)	Transactions
RosettaNet	Partner Interface Process (PIP), Business Process
EbXML (Electronic Business Using XML)	Business Collaboration, Business Process
WSCL (Web Services Conversation Language)	Conversation
WSCI (Web Services Choreography Interface)	Process, Collaboration
BPML (Business Process Modeling Language)	Process
BPEL4WS (Business Process Execution) Language for Web Services	Abstract Process, Executable Process

Table 1 • E-business dialogues

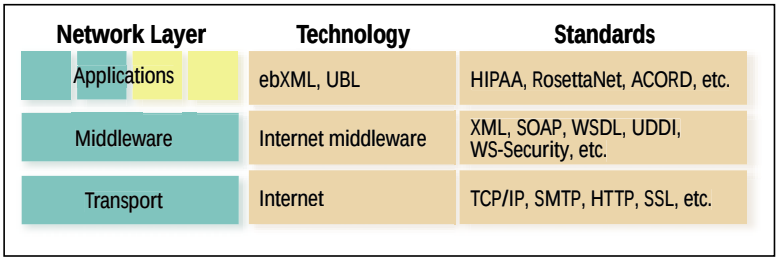


Figure 2 • Web services standards are Internet middleware (Forrester Research, Inc.)

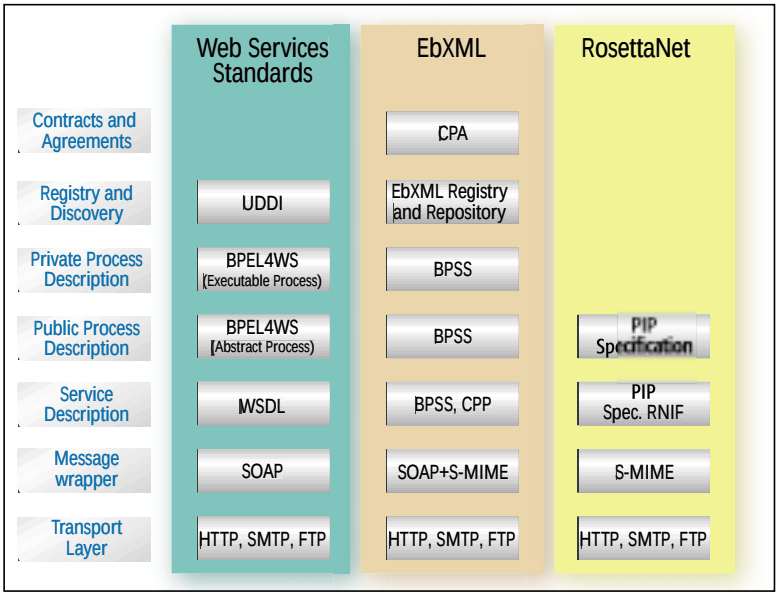


Figure 3 • Understanding how RosettaNet, EbXML, and BPEL stack up

of momentum; it is the focus of the rest of this article.

To conduct an e-business dialogue using the WSDL and BPEL combination, the partners must have a Web services framework implemented. They could then construct a mutually agreeable e-business dialogue definition using the abstract process feature in BPEL. The partners can reuse existing Web services for this e-business dialogue, and create new Web services where necessary. Currently, the partners would have to define everything in the dialogue, including the vocabulary used, the messages exchanged, the choreography of the messages, and the definition of the business process. Hopefully, in the near future partners won't have to keep reinventing the wheel and can benefit from standardized processes that include standard vocabulary, messages, and choreography. This series of articles demonstrates how partners can utilize standardized processes defined by RosettaNet when defining Web services with WSDL and BPEL.

Exploring BPEL

BPEL can be used to create e-business dialogues by "composing" new and previously defined Web services. BPEL is the marriage of two rival specifications: WSFL from IBM, and XLANG from Microsoft. BPEL comes with a few complementary specifications, WS-Transaction and WS-Coordination, that take the BPEL process further by adding more transaction support and enabling processes to span different platforms, systems, and execution engines.

BPEL defines the choreography of Web services and the roles of the business partners involved in the process, and may include advanced features like exception and compensation handling for actions already performed. BPEL processes interact with new and previously defined Web services on any platform running within, or outside, a BPEL environment.

Two types of processes are defined in BPEL: abstract processes, which are the equivalent of defining public processes in an e-business dialogue, and executable processes, which run within the organization. The main difference between the two is that abstract processes cannot be executed; they are utilized as an agreement between two partners about how they will interact in the e-business dialogue. The abstract processes only define public processes and do not include what each organization does internally to conduct those public processes.

The executable process, on the other hand, is designed for executing business processes within an organization. It can link several Web services together, tie in business logic, and access sensitive data to achieve a business goal for the organization.

BPEL components

Figure 4 shows the process element definition in BPEL from the schema design view of XMLSPY. I am building the BPEL abstract process definition for a purchase order e-business dialogue and will be reusing WSDL definitions of the RosettaNet PIP3A4 defined in Part 1. I will explain BPEL in more detail as I build the BPEL definition of the e-business dialogue. The first element I have to define is the root element <process>:

```
<process name="loanApprovalProcess"
  targetNamespace="http://DifferentThinking.com/
   BPEL4WS/POProcess"
xmlns="http://schemas.xmlsoap.org/ws/2002/07/
  business-process/"
xmlns:PO="http://DifferentThinking.com/wsdl/
  PurchaseOrder/"
```

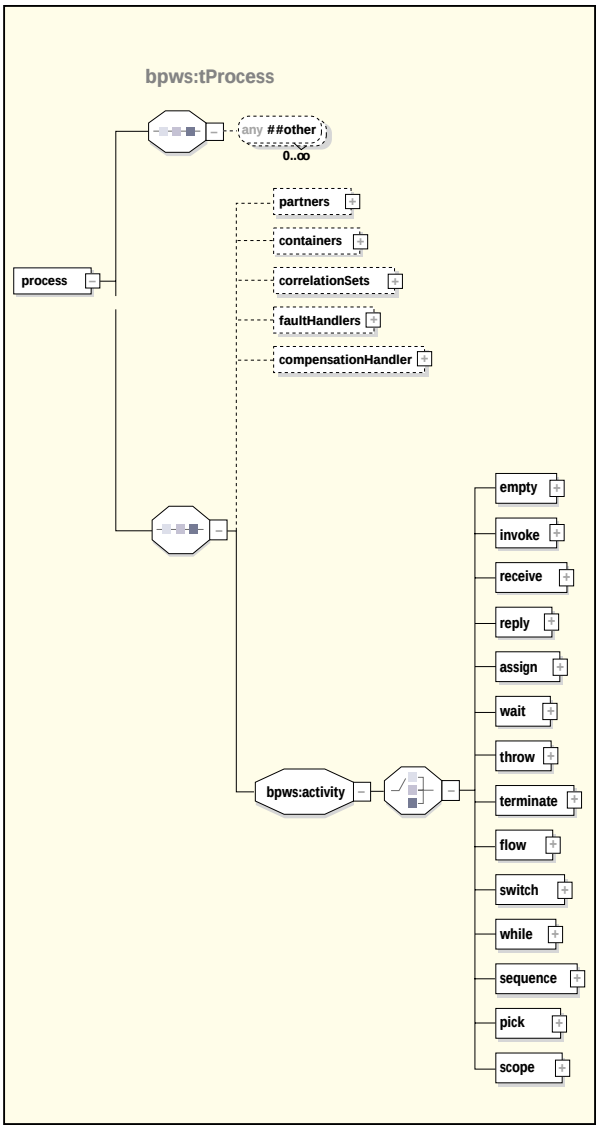


Figure 4 • Process element definition

```
abstractProcess="yes">
```

This process is the definition of the e-business dialogue between two business partners; this definition deals only with the public interactions between the two partners and not how they handle the business inside their organizations. This type of definition in BPEL is called an *abstract process*, and you can see that the attribute "abstractProcess" is set to "yes" to indicate this.

The process I'm currently describing is a synchronous process, assuming that when the buyer makes a purchase order request, the seller will be able to reply promptly with the answer. I will describe the more realistic, and slightly more involved, asynchronous process in Part 3, which focuses on creating and implementing more elaborate e-business dialogue definitions. Table 2 shows BPEL process components.

Partners

At a conceptual level, a BPEL process interacts with partners to receive, invoke, and reply to activities. These partners can be Web services within the organization or Web services at the business partner's organization. A process definition begins with describing which parties (Web services) are involved and what role they play. The most important part of

the partner description is the service link type; it is a mechanism used to tie the partner description to the operations the partners can perform from the WSDL definition.

To begin the partner definition in my BPEL process, I first need to add a service link type to the WSDL definition I defined in the previous article. I need to add the following to the <wsdl:definitions> element:

```
xmlns:slnk="http://schemas.xmlsoap.org/ws/2002/06/
 service-link/".
```

The next step is to add the service link type as an immediate child of the <wsdl:definitions> element:

```
<slnk:serviceLinkType name="purchaseOrderRequestLT">
  <slnk:role name="seller">
   <portType name="QuoteAndOrderEntryPort"/>
  </slnk:role>
</slnk:serviceLinkType>
```

Having updated the WSDL definition in the QuoteAndOrderEntry.wsdl file, I can start adding to my BPEL definition by using the service link defined above to compose a partner definition as follows (source code for PurchaseOrderProcess.bpel and QuoteAndEntry.wsdl is available for download at www.sys-con.com/xml/sourcec.cfm):

```
<partners>
  <partner name="buyer"
    serviceLinkType="PO:purchaseOrderRequestLT"
    myRole="seller"/>
  <partner name="seller"
    serviceLinkType="PO:purchaseOrderRequestLT"
    partnerRole="seller"/>
</partners>
```

This definition is straightforward. There are two partners to the process: the customer and another internal Web service acting as a seller to the process. The first partner definition shows that the process is a seller and offers the customer the functionality defined in the QuoteAndOrderEntryPort. The second partner definition shows that the process is dealing with a partner (the internal Web service), which is the seller to the process.

Containers

The BPEL process uses containers to store state information such as temporary variables or incoming and outgoing messages. A container is really a WSDL message type and can be defined as an input or output container within invoke, receive, and reply activities. Here is the definition of the container for the process I'm defining:

```
<containers>
 <container name="requestPO" messageType="PO:placePuchase
   OrderRequest"/>
 <container name="processedPO" messageType="PO:sendPur
  chaseOrderConfirmation"/>
</containers>
```

Correlation sets

A correlation set makes it possible for messages to be addressed to the correct port Type, and more importantly, the correct instance of the process. A partner may be executing various processes with several partners, and it's important that the message exchanged within the processes go not only to the right partner and the right port Type, but also to the precise

Process Component	Description
Partners	A partner can be a business partner's Web service, or a Web service within the organization
Containers	A container allows processes to store data, enabling stateful interaction (specialized cookies)
Correlations Sets	Relate messages to port types and process instances
Fault Handlers	Define how to handle exceptions in a process
Compensation Handlers	Define how to "undo" actions already executed
Activities	Define the actions used in a process

Table 2 • BPEL Process components

instance of the business process, since the partner might have instantiated several instances of the process. Correlation is an advanced feature for this installment and will be addressed in the next part of the series.

Fault handlers, compensation handlers, and scope

When a fault occurs at a WSDL port, it may fire off a fault message. This fault message reaches the BPEL process, which uses fault handlers to determine how to recover from the fault. One possible recovery option is to "undo" what the process has done so far. This recovery depends on how far the process had progressed when the fault occurred, and the best way to determine that is to use the scope structured activity.

The scope activity defines fault handlers for a collection of activities. Each activity is nested within a catch clause, and if a fault occurs, a fault handler for the activity handles the fault. The fault handler uses compensation handlers to undo actions that have already been committed. The scope activity ensures that either all activities defined within the scope will complete successfully, or all activities are "compensated."

To enable compensation handlers, partners need to provide an "undo" operation for every "do" operation. For example, if an operation from SafeAirlines allows "BuyTicket", SafeAirlines must also provide a "CancelTicket" operation. I'm not using fault handlers or compensation handlers for the process in this installment, so it's time to move on to activities, where all the action takes place.

Activities

A BPEL process uses activities to choreograph Web services into longer-running e-business dialogues.

BPEL provides a collection of basic and structured activities to use in a process. The difference between the two types of activities is that a basic activity cannot enclose other activities, while a structured activity encloses a collection of activities. The basic activities are: empty, invoke, receive, reply, assign, wait, throw, and terminate. Structured activities are: flow, switch, while, sequence, pick, and scope.

The process interacts with other Web services with invoke, receive, and reply activities, and these are the activities we will use in the current installment of the article. A BPEL process allows only one activity per process, so if you need to conduct a collection of activities, you can use a structured activity to enclose the collection. I will use the sequence activity to provide a sequential order to conduct the activities. The e-business dialogue described here receives a purchase order request from a buyer and replies to the buyer with a purchase order confirmation.

```
<sequence>
 <receive name="PORequest" partner="buyer"
```

```
  portType="PO:QuoteAndOrderEntryPort"
  operation="requestPurchaseOrder" container="requestPO"
  createInstance="yes">
 </receive>
```

The definition above tells the process to monitor the given port Type for the specified operation conducted by the described partner. When the process receives a message matching these three criteria, it searches for the correct active "receive activity" and passes the received message to it. In our case, the receive activity stores the message in the requestPO container and ends execution. The sequence activity passes the execution control to the next activity in the sequence, the invoke activity. A BPEL process can be created only if the first activity in the process is the receive activity with the createInstance attribute set to "yes".

The process now moves to the next step – invoking an internal Web service with the inventory department, which goes over the purchase order request item by item, marking each with a status of "accepted", "rejected", or "pending". The inventory Web service receives the purchase order it needs to process through an input container and it places the processed purchase order in the output container. Since the workings of this Web service are not a public process, the details on how it works are not provided in the abstract process. Here is how the invoke activity is described:

```
<invoke name="POApprover"
  partner="seller"
  portType="PO:QuoteAndOrderEntryPort"
  operation="requestPurchaseOrder"
  inputContainer="requestPO"
  outputContainer="processedPO">
 </invoke>
```

The next step is to add a reply to the sequence activity definition. The process receives the processed purchase order from the inventory Web service through the processedPO container. It replies to the buyer's request by sending the buyer the processed purchase order. This is the definition for the reply:

```
 <reply name="POResponse" partner="buyer"
   portType="PO:QuoteAndOrderEntryPort"
   operation="requestPurchaseOrder" container="processedPO">
 </reply>
    </sequence>
</process>
```

Conclusion

We now have a simple BPEL definition of a RosettaNet PIP. However, we are just getting started…Part 3 is all about implementation. I'll build detailed abstract and executable asynchronous processes describing entire business scenarios, composed of several RosettaNet PIPs, and show you how to implement the BPEL processes using BPWS4J. Part 3 implements all the concepts and definitions covered in Parts 1 and 2. It promises to be an exciting ride!

Author Bio

Suhayl Masud is the founder and lead consultant at Different Thinking, a consulting firm that enables organizations to conduct electronic business by providing training, architecture, and application construction services. Suhayl's experience includes consulting as the lead technical architect for RosettaNet, where he helped define the next generation of e-business process standards.

SUHAYL@DIFFERENTTHINKING.COM

WRITTEN BY **YUHANG SUN**

Developing Complex XSLT Scripts

Design patterns address flexibility and scalability

XSLT is a declarative language designed for transforming XML documents into documents in any format. In developing large-scale XSLT scripts, software qualities such as flexibility and maintainability become issues. To address those issues, this article will first discuss a design approach that emphasizes a decomposition and recomposition view of transformations; then, a technique for modularizing XSLT scripts will be described; and finally, a few design patterns will be introduced.

Identify Responsibilities and Assign Them to Code Units

The goal of any kind of application design is to divide responsibilities into smaller pieces and conquer them in separated code units. In XSLT, call-by-name templates, call-by-context templates, and global variables are code units, which are equivalent to functions in structured programming languages. Global variables are equivalent to call-by-name templates without any parameters. Call-by-context templates require callers and callees to share context information. Examples of three types of code units are shown in Listing 1.

Having identified code units in XSLT, we should study how to divide a complex transformation into smaller ones and assign each to a code unit.

At the most abstract level, due to the stateless nature of XSLT, the responsibility of any XSLT script can be described as transforming a set of input streams to an output stream. Intuitively, outputting a segment of the output stream can be the candidate of a smaller responsibility.

Outputting each line in Listing 2 could be a responsibility. However, abstraction is required to identify reusable and semantically significant segments. Identifying those segments, which may not even be continuous, requires insight into the output stream. It's easy for a C programmer to find outputting the conditional expressions a meaningful responsibility. The template that fulfills this responsibility is implemented in Listing 3.

Identifying smaller responsibilities is a decomposition process, which decomposes an output stream into pieces. Consequently, the assembly or recomposition of those smaller segments into the output stream is necessary. The template in Listing 4 assembles the outputs from the template in Listing 3 and outputs the C++ code in Listing 2. An example XML context while calling the template in Listing 4 is also listed.

Decomposition and recomposition are not only design techniques, they're also an accurate vocabulary for describing the functionality of a code unit. Using this approach, it's important to constantly consider what a code unit does from the perspective of decomposition and recomposition. For example, the template in Listing 3 outputs conditional expressions, which are segments of the implementations of equality operators. The template can be reused elsewhere because it abstracts a common concept. The template in Listing 4 outputs the implementations of equality operators by assembling and decorating the outputs from the template in Listing 3. Figure 1 shows the composite relationship of the templates in Listings 3 and 4 and some other templates, which is a good way to show design and implementation structure.

The use of decomposition and recomposition in XSLT is very much like the process of functional decomposition in structured programming, which produces finer-grained code units calling each other hierarchically.

Modularize Code Units into Files

Grouping code units into coarse-grained code modules is a basic technique for managing complexity. In XSLT, there are no concepts such as classes (in object-oriented programming) to use for grouping code units, though files provide natural boundaries. Given a stylesheet file, it's crucial for complexity management to be able to illustrate its grouping criterion. The criterion should be recorded in the leading comment of the file. The grouping criterion of a stylesheet file summarizes the characteristics of the code units that are both contained in the file and called from other files. Those code units will be referred as public code units as opposed to the ones used internally.

The commonalities of the input or output of code units are intuitive and effective criteria for grouping. Figure 2 shows the stylesheet files used in a project and their interdependencies. The code units in "class-wrapper.xsl" take "class.xml" as their inputs; the code units in "intermediate-wrapper.xsl" call the public code units of "class-wrapper.xsl" and "datatype-wrapper.xml"; "c-source-file.xsl" contains the code unit to assemble the segments outputted from the public code units in "intermediate-wrapper.xsl" and outputs a complete C++ source file, "class.cpp". "class-wrapper.xsl" and "intermediate-wrapper.xsl" use input commonalities as their grouping criteria, while "c-source-file.xsl" uses output commonality. This grouping approach, along with the decomposition and recomposition view of transfor-

AUTHOR BIO

Yuhang Sun is a software engineer for Object Computing, Inc., in St.Louis. He is the major XML/XSLT developer for a project that is very successful in applying XML-related technologies.His interests also include j2EE and .NET. He holds a bachelor's degree in electrical engineering from TsingHua University in Beijing and a master's degree in computer science from Washington University.

mations, results in traceable code where, given a responsibility, locating the implementing code unit is easy: in the vocabulary of decomposition and recomposition, the input and output of the code unit should be clear. According to the input and output, the stylesheet file containing the implementation of the code unit can be identified. At last, the code unit can be located by reading through the identified stylesheet file. This last step may take a long time if the file is large. The step can be made fast by grouping code units within the file and systematically commenting. Listing 5 shows the content of "intermediate-wrapper.xsl", whose public code units are grouped by the file they will be called from. The comments starting with "Group" denote the boundary between those groups.

Since there is no mechanism to mark a code unit as public, a naming convention that prefixes the names of public code units with the names of their containing stylesheet files is used. For example, the names of templates in Listing 5 are prefixed with "intermediate-wrapper". The naming convention makes public code units prominent; moreover, it specifies their implementing file, which makes the code more traceable.

Grouping is for the purpose of organizing code units and encapsulating them. Besides being the physical shelves of code units, stylesheet files are also units of encapsulation. A stylesheet file exposes its functionalities through public code units. Each public unit should fulfill a well-defined responsibility in terms of decomposition and recomposition. Don't expose private data through any public code unit. Call-by-context templates require callers and callees to share context information and so tend to break encapsulation. Therefore, only call-by-name templates and global variables should be used for public code units.

Applying Design Patterns

Grouping code units into files based on the commonalities of their input and output is effective in organizing code. However, the application of fundamental software engineering principles and design patterns plays the key role in managing complexity. Both patterns described next are typical and may apply to a variety of contexts.

Intermediate XML Tree

Instead of directly transforming input documents to output documents, transform the input documents to some intermediate XML trees and then transform the intermediate XML trees to the output documents.

PROBLEM

Direct transformation of input documents to output documents is so complex that it may cause the use of convoluted XPath expressions. Using this pattern, the original transformation is divided into two simpler subtransforms; one takes the input documents and creates the intermediate XML trees, while the other takes the intermediate XML trees and generates the output documents.

Another problem is that the formats of input and output documents keep changing throughout the stylesheet development, even though those documents convey relatively stable semantics. With direct transformation, the transforming XSLT scripts need significant modification according to the changes. Using this pattern, the intermediate XML trees convey the stable semantics with stable formats. Because the formats of the intermediate trees are relatively stable, there is less coupling between subtransformations (see Listing 6).

The logical mode in Listing 6 defines two data types, Alphanumeric and LastName. Data type LastName is derived from Alphanumeric, adding a max-length facet. The logical model clearly describes the semantics of data types but it could not be used easily for generating code. The intermediate tree, which can be transformed from the logical model, includes a semantics element for each data-type element, which can be directly used for generating code. The data types with unbounded string semantics could be mapped to the pointer type in C or the String class in Java, while the data type with bounded string semantics could be mapped to arrays in both languages.

Imagine that the structure of the logical model in Listing 6 changes to the structure in Listing 7.

The two logical models convey exactly the same information with different formats. Due to its abstractness, the intermediate tree will remain unchanged. Therefore, the code depending on the intermediate tree is not affected.

DISCUSSION

The goal of using intermediate XML trees is to reduce the complexity and impact of the changes of input or output documents. The intermediate XML trees must reflect the stable semantics of input and output documents with stable structures; moreover, the structures of intermediate XML trees should be easier to use for generating final outputs. Intermediate XML trees are more useful than temporary structures and often reflect the most important design abstractions.

Builder

Separate responsibility of the assembly of a complex output from its representation. A director template is responsible for parsing and assembling. Different builder templates, which are invoked by the director template as callback, are responsible for creating different representations.

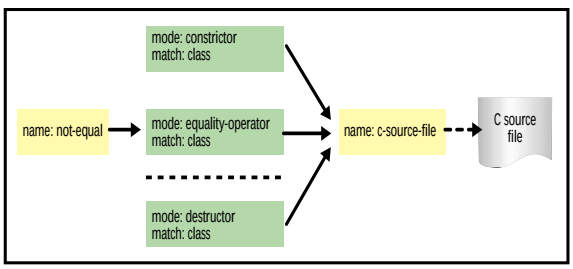


Figure 1 • Composite relationships among templates

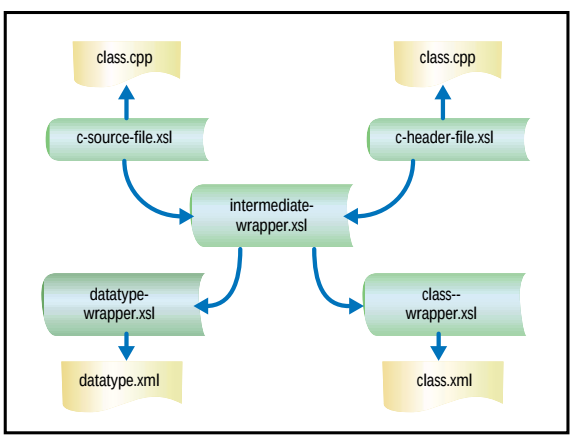


Figure 2 • Stylesheet files, input and output documents, and their interdependencies

PROBLEM

An input document needs to be transformed to multiple output documents in different formats. The transformation to each format calls for the same parsing and assembling process. Using this pattern, the director template encapsulates the parsing and assembling process. Therefore, the builder templates can focus on the representations of their output documents without knowing the assembling process and

depending on the structure of the input document.

Following the builder pattern, the code in Listing 10 realizes the transformation from the input document in Listing 8 to the output documents in Listing 9.

While invoking the director template in Listing 10, the element node "tp:marshalling", which uniquely identifies the callback builder template for marshalling, should be passed in. Refer to "The Functional Programming Language XSLT - A proof through examples" on how to treat templates as first-class data. The director template invokes the callback builder templates at certain points during parsing of the input document. Both callback builder templates take a parameter, "style", which is set to either "optional" or "field" depending on the parsing contexts where the templates are invoked. The callback builder templates output C++ instructions according to the value of this parameter.

DISCUSSION

This pattern is very similar to the object-oriented Builder pattern discussed in "Design Patterns: Elements of Reusable Object-Oriented Software." The ability to treat templates as first-class data types makes some of the techniques used in object-oriented or functional programming applicable to XSLT programming.

Summary

The decomposition and recomposition view of XSLT scripts is essential. It is always possible to describe a transformation with steps of decomposition and recomposition, no matter what advanced design patterns are used. XSLT lacks the entities for modularization and encapsulation. The commonalities of inputs and outputs of code units are intuitive criteria for modularization. The use of a few code and name conventions enforces modularization and encapsulation. Intermediate XML Tree is a powerful pattern for simplifying problems, which also results in codes that are easily adjustable to changes. The Builder pattern shows how to treat templates as first-class data types and how to borrow the ideas from the object-oriented world.

Acknowledgments

Special thanks to Mario Aquino for his review and excellent comments.

References

- Gamma, Erich; Helm, Richard; Johnson, Ralph; and Vlissides, John. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley.
- Novatchev, Dimitre. "The Functional Programming Language XSLT - A proof through examples": www.topxml.com/xsl/articles/fp/1.asp

SUNYUHANG@HOTMAIL.COM

LISTING 1 • Examples of three types of code units

```
An example of call-by-name templates:

The definition of a call-by-name template:
<xsl:template name="template-name">
<xsl:param name="parameter1"/>
<!-- the implementation of the template is here -->
</xsl:template>

The invocation of the call-by-name template:

<xsl:call-template name="template-name">
 <xsl:with-param name="parameter1" select="'value'"/>
</xsl:call-template>

An example of call-by-context templates:
The definition of a call-by-context template:

<xsl:template match="xml-node-name">
<xsl:para name="parameter1"/>
<!-- the implementation of the template is here -->
</xsl:template>

The invocation of the call-by-context templates:

<xsl:apply-templates select="context-node/xml-node-name">
 <xsl:with-param name="parameter1" select="'value'"/>
</xsl:apply-templates>

An example of global variables:
The definition of a global variable:

<xsl:variable name="global-variable">
  <!-- the implementation of the global variable is here -->
</xsl:variable>

The invocation of the global variable:

$global-variable
```

LISTING 2 • The C++ code to output

```
bool AClass::operator== ( const AClass & rhs ) const
{
if (intField1_!=rhs.intField1_)  return false;
if (intField2_!=rhs.intField2_)  return false;
if (strcmp(stringField_,rhs.stringField_) != 0)  return
false;
return true;
}
```

LISTING 3 • The code unit outputting conditional expressions

```
<xsl:template name="not-equal">
 <xsl:param name="left"/>
       <xsl:param name="right"/>
 <xsl:param name="type"/>
 <xsl:choose>
  <xsl:when test=" $type = 'int'">
   <xsl:value-of select="concat($left,'!=',$right)"/>
  </xsl:when>
<xsl:when test=" $type = 'char*'">
   <xsl:value-of
select="concat('strcmp(',$left,$right,')!=0)')"/>
  </xsl:when>
 </xsl:choose>
</xsl:template>
```

LISTING 4 • The template to recompose and an example calling context

```
An example calling context:

<class class-name="Aclass">
 <instance-variable variable-name="intField1_"  variable-
type="int"/>
 <instance-variable variable-name="intField2_"  variable-
type="int"/>
 <instance-variable variable-name="stringField_"  vari-
able-type="char*"/>
</class>

The assembling template:

<xsl:template mode="equality-operator" match="class">
 <xsl:value-of select="concat('bool ',@class-name,'::oper-
ator == (',@class-name,' &amp; rhs )
const',$cr,'{',$cr)"/>
 <xsl:for-each select="instance-variable">
  <xsl:variable name ="conditionalExpression">
   <xsl:call-template name="not-equal">
    <xsl:with-param name="left" select="@variable-name"/>
    <xsl:with-param name="right"
select="concat('rhs.',@variable-name)"/>
    <xsl:with-param name="type" select="@variable-type"/>
```

```
   </xsl:call-template>
  </xsl:variable>
  <xsl:value-of select="concat('if
('',$conditionalExpression,'') return
false;',$cr)"/>
 </xsl:for-each>
 <xsl:value-of select="concat('return
true;',$cr,'}',$cr)"/>
</xsl:template>
```

LISTING 5 • The content of intermediate-wrapper.xsl

```
<!-- Group: The following templates
are called from c-source-file.xsl
->

<!-- output the constructor of a class
->
<template name="intermediate-
wrapper.constructor">
 <xsl:param name="className"/>
……
</template>
……

<!-- Group: The following templates
are called from c-header-file.xsl
->
<!-- output the include section of a
class ->
<template name="intermediate-
wrapper.inlcudes">
 <xsl:param name="className"/>
……
</template>
……
```

LISTING 6 • The input XML document

```
<logical-model>
 <data-type>
  <name>Alphanumeric</name>
  <description> Alphanumeric represents
the character space [48..57, 65..90,
97..122] of the ASCII character
set</description>
 </data-type>
 <data-type>
  <name>LastName</name>
  <description> the last name of a
person can have up to 10 Alphanumeric
characters</description>
  <base-type>Alphanumeric</base-type>
  <max-length>10</max-length>
 </data-type>
</logical-model>

<intermediate-tree>
 <data-type>
  <name>Alphanumeric</name>
  <semantics>
   <unbounded-string/>
  </semantics>
 </data-type>
 <data-type>
  <name>LastName</name>
  <semantics>
   <bounded-string max-length="10"/>
  </semantics>
 </data-type>
</intermediate-tree>
```

LISTING 7 • Changed logical model

```
<logical-model>
 <data-type>
  <name>Alphanumeric</name>
  <description> Alphanumeric represents
the character space [48..57, 65..90,
97..122] of the ASCII character
set</description>
 </data-type>
 <data-type>
  <name>LastName</name>
  <description> the last name of a
person can have up to 10 Alphanumeric
characters</description>
  <base-type>Alphanumeric</base-type>
  <facet>
<value>10</value>
<facettype>
 <name>Max Length</name>
</facettype>
  </facet>
 </data-type>
</logical-model>
```

LISTING 8 • The input document

```
<class name="Object">
 <fields>
<optional flagName="optFlag">
   <fields>
    <field name="opt_F"/>
   </fields>
  </optional>
  <field name="nonOpt_F"/>
 </fields>
</class>
```

LISTING 9 • The two output documents

```
C++ Marshaling code segment:

 write(object.optFlag);
 if ( object.optFlag ) {
  write(object.opt1_F);
  delete object.opt1_F;
 }
 write(object.nonOpt_F);
 delete object.nonOpt_F;

C++ De-marshaling code segment:

 read(object.optFlat) {
 if ( object.optFlag ) {
  read(object.opt1_F);
 }
 read(object.nonOpt_F);
```

LISTING 10 • The sample implementation of builder patterns

The director templates:

```
<xsl:template mode="director"
match="class">
 <xsl:param name="callBack"/>
 <xsl:apply-templates mode="director"
match="fields/*">
  <xsl:with-param name="callback"
select="$callBack"/>
 </xsl:apply-templates>
</xsl:template>

<xsl:template mode="director"
match="optional">
<xsl:param name="callBack"/>
 <xsl:apply-templates mode="template-
pointer" select="$callBack">
  <xsl:with-param name="flagName"
select="@flagName"/>
  <xsl:with-param name="style"
select="'optional'"/>
 </xsl:apply-templates>
 <xsl:value-of select="'{'"/>
 <xsl:apply-template mode="director"
match="fields/*">
  <xsl:with-param name="callback"
select="$callBack"/>
 </xsl:apply-templates>
 <xsl:value-of select="'}'"/>
</xsl:template>

<xsl:template mode="director"
match="field">
<xsl:param name="callBack"/>
 <xsl:apply-templates mode="template-
pointer" select="$callBack">
  <xsl:with-param name="fieldName"
select="@name"/>
  <xsl:with-param name="style"
select="'field'"/>
 </xsl:apply-templates>
</xsl:template>
```

The callback builder templates:
The builder to output marshalling code:

```
<xsl:template mode="template-pointer"
match="tp:marshalling" xmlns:tp="tem-
plate-pointer-simulator-prefix">
 <xsl:param name="fieldName"/>
 <xsl:param name="flagName"/>
 <xsl:param name="style"/>
 <xsl:choose>
  <xsl:when test="$style = 'optional'">
   <xsl:value-of
select="concat('write(object.',$flagNam
e,');',$cr)"/>
   <xsl:value-of select="concat('if
(object.',$flagName,')',$cr)"/>
  </xsl:when>
  <xsl:when test="$style = 'field'">
             <xsl:value-of
select="concat('write(object.',$field-
Name,');',$cr)"/>
             <xsl:value-of
select="concat('delete object.',$field-
name,';',$cr)"/>
  </xsl:when>
 </xsl:choose>
</xsl:template>
```

The builder to output de-marshalling code:

```
<xsl:template mode="template-pointer"
match="tp:de-marshalling"
xmlns:tp="template-pointer-simulator-
prefix">
 <xsl:param name="fieldName"/>
 <xsl:param name="flagName"/>
 <xsl:param name="style"/>
 <xsl:choose>
  <xsl:when test="$style = 'optional'">
   <xsl:value-of
select="concat('read(object.',$flagName
,');',$cr)"/>
   <xsl:value-of select="concat('if
(object.',$flagName,')',$cr)"/>
  </xsl:when>
  <xsl:when test="$style = 'field'">
   <xsl:value-of
select="concat('read(object.',$field-
Name,');',$cr)"/>
  </xsl:when>
 </xsl:choose>
</xsl:template>
```

The XML file for simulating template pointers:

```
<model xmlns:tp="template-pointer-simu-
lator-prefix">
 <pointers>
  <tp:marshalling/>
  <tp:de-marshalling/>
 </pointers>
</model>
```

The invocation of the director template in order to output marshalling code:

```
<xsl:apply-templates mode="director"
select="class" xmlns:tp="template-
pointer-simulator-prefix">
 <xsl:with-param name="callback"
select="pointers/tp:marshalling"/>
</xsl:apply-templates>
```

Download the Code
www.sys-con.com/xml

THE LARGEST WEB SERVICES, JAVA, XML AND .NET CONFERENCE AND EXPO IN THE WORLD!

Boston 2003 | London 2003 | Berlin 2003 | Hong Kong 2003

JDJ EDGE conference&expo
web services EDGE conference&expo
.NET EDGE conference&expo
XML EDGE conference&expo

Register by February 14th SAVE UP TO $400
Register by March 14th SAVE UP TO $200

3rd Annual

Web Services Edge EAST Conference & Exposition

FINAL PROGRAM

Connecting the Enterprise with Web Services, Java, XML, & .NET

March 18–20, 2003
Hynes Convention Center
Boston

CALL TODAY TO REGISTER
201-802-3058
OR REGISTER ONLINE AT
www.sys-con.com

Event Sponsors

Microsoft · Sun microsystems · SYBASE · IONA · DIGITAL EVOLUTION · COMPUWARE · ORACLE · hp invent · Rational the e-development company · ALTOVA

Media Sponsors

JAVA DEVELOPER'S JOURNAL · JavaWorld · WebServices JOURNAL · LINUX BUSINESS WEEK · XML JOURNAL · SAMS · BASIS · Perl.com DEVMAG · .NET DEVELOPER'S JOURNAL · SD Times · asp.netPRO · WebSphere DEVELOPER'S JOURNAL · WebLogic · wireless · CF Advisor · ColdFusion Developer's Journal · PowerBuilder Developer's Journal

OWNED BY SYS-CON MEDIA

PRODUCED BY SYS-CON EVENTS

Special Insert: Web Services Edge East Conference & Expo

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

Special Insert: Web Services Edge East Conference & Expo

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

web services EDGE conference&expo

JDJ EDGE conference&expo

.NET EDGE conference&expo

XML EDGE conference&expo

THE LARGEST WEB SERVICES CONFERENCE & EXPO

FEATURES & ATTRACTIONS

- **3 Days Packed with Education and Training**
- **4 Keynotes & 3 Panel Discussions**
- **60 Hard-Hitting and Current Seminars**
- **FREE .NET Tutorial with Russ' Tool Shed**
- **Web Services & XML Tutorials**
- **Java University Certification Training**
- **HOT** Industry-Leading Certification Programs
- **1 Day Mobile .NET Tutorial**
- **INFORMATIVE** Round Table Discussions
- Opening Day Welcome Reception
- SAMS Meet the Authors Hot Topics Lounge
- **COMPELLING** Case Studies & Best Practices
- **FEATURED** Product Demonstrations on the Show Floor
- **RIVETING** Real-time SYS-CON Radio Interviews

WHO SHOULD ATTEND

- **Software Developer**
- **Software Engineer**
- **Development Manager**
- **Application Developer**
- **Technical Director**
- **Analyst/Programmer**
- **IT Manager**
- **Technical Architect**
- **Team Leader**
- **Software Consultant**

Keynotes & Highlighted Speakers

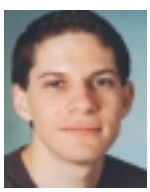
KEYNOTE SPEAKERS

John Magee
Vice President, Oracle9i

John Magee is vice president, Oracle9*i,* at Oracle. He has more than 14 years' experience in the enterprise software industry and has held positions in product development, product management, and product marketing. In his current role, Magee manages technical product marketing for Oracle's application server and development tools products, and is responsible for evangelizing Oracle technology initiatives around J2EE, XML, and Web services.

Mark Herring
Director Java, Web Services & Tools Business

Mark Herring is responsible for helping to define, set, and drive Sun Microsystems' product direction in the Java, Web Services & Tools Business. Prior to his current role, Herring was director of corporate strategy & planning, looking after Sun's interest in the Project Liberty Alliance and Network Identity. Herring joined Sun Microsystems in October 1999 as a result of Sun's acquisition of Forte Software. Forte Software was a leading provider of enterprise-class development and integration products. During his four years at the company, he ran several aspects of Forte's marketing organization, including product marketing and the Web channel.



Miguel de Icaza
Cofounder and CTO

As the founder and leader of the GNOME Foundation, Miguel de Icaza is one of the foremost luminaries in the Linux development community. With his seemingly boundless energy, de Icaza has galvanized the effort to make Linux accessible and available to the average computer user. He brings this same excitement to his role as CTO of Ximian. de Icaza was instrumental in porting Linux to the SPARC architecture and led development of the Midnight Commander file manager and the Gnumeric spreadsheet. He is also a primary author of the design of the Bonobo component model, which leads the way in the development of large-scale applications in GNOME.

Mark Hapner
Distinguished Engineer, Sun Microsystems

Mark Hapner is a Sun Distinguished Engineer and is currently lead architect for Java™ 2 Platform, Enterprise Edition (J2EE™). He has guided the overall architecture for J2EE 1.2, 1.3, and now the upcoming 1.4 release. In March of 1996, he joined Sun's JavaSoftware Division to participate in the development of the Java database connectivity API (JDBC). Following that, he was co-spec lead of the Enterprise JavaBeans specification and spec lead of the Java Message Service specification.

Simon Phipps
Chief Technology Evangelist, Sun Microsystems

Simon Phipps, currently chief technology evangelist at Sun Microsystems, speaks frequently at industry conferences on the subject of technology trends and futures. He was previously involved in OSI standards in the 1980s, in the earliest collaborative conferencing software in the early 1990s, and in introducing Java and XML to IBM.

Dave Chappell
VP, Chief Technology Evangelist, Sonic Software

Dave Chappell is the vice president and chief technology evangelist for Sonic Software. He has more than 18 years of industry experience building software tools and infrastructure for application developers, spanning all aspects of R&D, sales, marketing, and support services. Chappell has published in numerous technical journals, and is currently writing a series of contributed articles for *Java Developer's Journal*.

Eric Newcomer
Chief Technology Officer, IONA

In the role of chief technology officer at IONA, Eric Newcomer is responsible for IONA's technology roadmap and the direction of IONA's Orbix E2A e-Business Platforms as relates to standards adoption, architecture, and product design. Newcomer joined IONA in November 1999, and most recently served as IONA's vice president of engineering, Web Services Integration Products. He is a member of the XML Protocols and Web Services Architecture working groups at the W3C and IONA's Advisory Committee representative to UDDI.org.

Dean Guida
CEO and President, Infragistics

Dean Guida is CEO and president of Infragistics and was CEO and a cofounder of ProtoView Development Corporation. Mr. Guida has over 15 years of experience in the technical industry and oversees all aspects of the company's business operations and corporate direction. He is also responsible for cultivating strategic alliances and other external relationships, as well as managing corporate financial affairs.

Conference Advisory Board

Sean Rhody
Editor in Chief,
Web Services Journal
Partner, CSC

Alan Williamson
Editor in Chief,
Java Developer's Journal
Chief Technology Officer, n-ary

Derek Ferguson
Editor in Chief,
.NET Developer's Journal
Chief Technology Evangelist, Expand Beyond Corp.

Bob Familiar
.NET Architect,
Microsoft
New England

Thomas Robbins
Senior Technology Specialist,
Microsoft
New England

Hitesh Seth
Editor in Chief,
XML-Journal
Chief Technology Officer, ikigo

J.P. Morgenthal
Chief Services Architect, Software AG

Special Insert: Web Services Edge East Conference & Expo

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

Conference at-a-Glance

THE LARGEST AND THE MOST

		JAVA	WEB SERVICES	.NET
TUESDAY MARCH 18	8:00AM — 4:00PM	Registration Open		
	9:00AM — 9:50AM	(JV1) Squeezing the Best Out of Java Alan Williamson, Java Developer s Journal	(WS1) Web Services Infrastructure Carl Sjogreen, BEA	(NT1) .NET Framework Overview Bob Familiar, Microsoft
	10:00AM — 10:50AM	Web Services Keynote: John Magee, Oracle		
	11:00AM — 11:50AM	(JV2) Testing Your Java Using JUnit Kyle Gabhart, LearningPatterns	(WS2) Web Services Management James Phillips, Actional	(NT2) Introduction to ASP.NET Russ Fustino, Microsoft
	1:00PM — 1:50PM	WS-I Panel: A Road Map for Web Services Standards - Moderated by Rob Cheng, WS-I		
	2:00PM — 2:50PM	.NET Keynote: The MONO Project - Miguel de Icaza, Ximian		
	3:00PM — 3:50PM	(JV3) Building/Deploying the Ant Way Kyle Gabhart, LearningPatterns	(WS3) Strategies for Using Databases in a World of Web Services Mike Lehmann, Oracle	(NT3) Introduction to VB.NET Russ Fustino, Microsoft
	4:00PM — 4:50PM	(JV4) Unlocking the Secrets of JDK1.4 Raghavan Srinivas, Sun Microsystems	(WS4) Using Web Services to Integrate J2EE and .NET Enterprise Applications - Odysseas Pentakalos, SYSNET International	(NT4) How to Develop an End-to End .NET Connected Appilication Allan da Costa Pinto, Microsoft
WEDNESDAY MARCH 19	8:00AM — 4:00PM	Registration Open		
	9:00AM — 9:50AM	(JV5) Java APIs for Web Services Security Standards Sang Shin, Sun Microsystems	(WS5) Combining BPM and BRM Technologies: A Major Step Towards Corporate Agility Henry Bowers, ILOG	(NT5) .NET: The Virtualized Execution Engine Yahya Mirz, Aurora Borealis
	10:00AM — 10:50AM	Java Keynote: Mark Herring, Sun Microsystems		
	11:00AM — 6:00PM	EXPO OPEN 11:00 a.m. - 6:00 p.m.		
	11:00AM — 11:50AM	(JV6) To Not Swing Is to SWT! The Swing Alternative - IBM	(WS6) Web Services for Real-Time Data Access in an Industrial Setting Stephan Van Dijck, ABB/SKYVA	(NT6) Introduction to DotGNU Barry Fitzgerald, DotGNU
	12:00PM — 2:00PM	BREAK & EXPO		
	2:00PM — 2:50PM	.NET Panel Discussion - Moderated by Derek Ferguson, .NET Developer s Journal		
	3:00PM — 3:50PM	(JV7) Unlocking the Power of XML Hitesh Seth, ikigo	(WS7) Web Services Architecture: The Next Big Spec. from the Mouths of the W3C Eric Newcomer, IONA (moderator)	(NT7) Introduction to SSCLI Yahya Mirz, Aurora Borealis
	4:00PM — 4:50PM	(JV8) Java and .NET Derek Ferguson, Expand Beyond	(WS8) Web Services: Next Steps After the Hype Claire Dessaux, Oracle	(NT8) Mobile Development with the Compact Framework Brad Mccabe, Infragistics
THURSDAY MARCH 20	8:00AM — 4:00PM	Registration Open		
	9:00AM — 9:50AM	(JV9) Writing SOAP Services Nigel Thomas, SpiritSoft	(WS9) Web Services Best Practices Chris Peltz, HP	(NT9) Best Practices for .NET Develop ment Joe Stagner, Microsoft
	10:00AM — 10:50AM	.NET Keynote - Jesse Liberty, Liberty Associates		
	11:00AM — 4:00PM	EXPO OPEN 11:00 a.m. - 4:00 p.m.		
	11:00AM — 11:50AM	(JV10) Working with Data the JDO Way Patrick Linsky, SolarMetric	(WS10) Web Services Startups: Telltales of the Future Simeon Simeonov, Polaris Venture Partners	(NT10) Best Practices for ADO.NET Development Thom Robbins, Microsoft
	12:00PM — 2:00PM	BREAK & EXPO		
	2:00PM — 2:50PM	Java Panel - The Future of Java , Moderated by Alan Williamson, Java Developer s Journal		
	3:00PM — 3:50PM	(JV11) Enterprise: The Next Generation Mark Hapner, Sun Microsystems	(WS11) Open Standards for Web Services Messaging Dave Chappell, Sonic Software	(NT11) How to Debug with .NET Tony Denbow, STAR Information Tech- nology
	4:00PM — 4:50PM	(JV12) Overcoming the Challenges of J2ME Dr. Jeff Capone, Aligo	(WS12) Web Services Security Marc Chanliau, Netegrity	(NT12) XML and Web-Enabling Legacy Applications Using BizTalk Mike Cramer, Microsoft

XML	VENDOR
(XM1) XML - A Manager s Guide JP Morgenthal, Software AG	Visit www.sys-con.com for details
(XM2) OASIS Standards Update Karl Best, OASIS	(VN2) The XMLSPY 5 Enterprise Edition Development Environment Trace Galloway, Altova
(XM3) A Definitive Introduction to XML Schemas Aaron Skonnard, DevelopMentor	(VN3) SOAP and Java: Marrying Them Off Skip Marler, Parasoft
(XM4) XML in Print - XSL:FO Frank Neugebauer, IBM	Visit www.sys-con.com for details
(XM5) XML Security Integration Challenges Phil Steitz, American Express	(VN5) Process-Centric Enterprises Eric Pulier, Digital Evolution
(XM6) Case Study: XML in Life Sciences Tim Matthews, Ipedo	(VN6) Pattern Driven Application Development Tom Shore, Compuware
(XM7) Using XML for EAI - Best Practices Dan Enache, TIBCO	(VN7) Managing the Developer Relationship Mike Bellissimo, Sun Microsystems
(XM8) Take XML with You: XML and Mobile Computing - Hitesh Seth, ikigo	(VN8) Web Services Diagnostics Dave Seidel, Mindreef
(XM9) XML, Ontologies and the Semantic Web - Ayesha Malik, Object Machines	Visit www.sys-con.com for details
(XM10) X Query Mike Champion, Software AG	(VN10) Model Driven Development of Web Services in UML for the J2ME Bill Graham, Rational Software
(XM11) XPath & XSLT 2.0 BEA Kurt Cagle, Cagle Communications	(VN11) Why Web Services Management? Jon Atkins, HP
(XM12) Third Generation XML Tools Michael Leventhal, Commerce One	Visit www.sys-con.com for details

JAVA UNIVERSITY℠ PROGRAM

Sun microsystems

9:00AM —5:00PM

Web Services Programming Using Java™ Technology and XML

This one-day seminar provides in-depth knowledge on Web services and shows how to develop Web services using the Java programming language and XML, the technologies of portable code and portable data respectively.

Sun microsystems

9:00AM —5:00PM

Java 2 Platform Programmer Certification Fast Path

This session, developed and delivered by Philip Heller, author of the two leading Java technology certification preparation manuals, helps to prepare you for the Sun Certified Programmer for the Java 2 Platform exam. Philip provides code-level, detailed review of the skills and knowledge needed to confidently approach the exam.

Sun microsystems

9:00AM —5:00PM

Java 2 Platform Architect Certification Fast Path

This intense one-day session helps prepare attendees to pass the Sun Certified Enterprise Architect for J2EE Technology exam. This session provides an overview of the components comprising the J2EE architecture as a whole, emphasizes the incorporation of J2EE technology into an architecture, and reviews each of the certification exam s testing objectives.

FAST TRACKS & TUTORIALS

XML EDGE conference&expo

9:00AM —5:00PM

XML Certified Developer Fast Path

This tutorial is for programmers who have some knowledge of XML and related technologies and would like to pass the IBM Certified Developer Test 141 on XML and Related Technologies.

Microsoft®

9:00AM —5:00PM

Russ Tool Shed

Join Russ as he shows you how to use Visual Studio.NET.

9:00-12:15 - Introduction to Web Services Using VS.NET

1:00-2:30 - Advanced Web Services Using ASP.NET

2:45-4:15 - .NET Remoting Essentials

RUSS' TOOLSHED TINKERING WITH VISUAL STUDIO

Microsoft .net

9:00AM —5:00PM

Mobile .NET

In this session, Derek Ferguson, editor-in-chief of .NET Developer s Journal, will give you a thorough introduction to the use of .NET with all manner of mobile computing devices.

Special Insert: Web Services Edge East Conference & Expo

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

SUN MICROSYSTEMS Java University℠ Program

Java University℠ Program: Aggressive, code-level training courses for experienced developers using Java™ technology, brought to you by Sun Microsystems, Inc. Attend seminars designed by industry luminaries and recognized experts. Sessions cover Sun certification and Web services technology. Whether you're a beginner or a veteran developer, architect, or software engineer, you'll benefit from these value-packed full-day courses. Register now. Seating is limited.

The Java University℠ program complements this year's Web Services Edge conference by offering 3 full-day training lectures for experienced software developers, architects, and engineers.

Java University℠ Program Take-Aways:

- Training designed and presented by expert Java technology engineers
- Student guides full of source code, examples, references and copies of instructors' materials
- Free Web-based training courses from Sun Educational Services

Tuesday, March 18, 2003

Web Services Using Java™ Technology and XML

SANG SHIN, SUN MICROSYSTEMS, INC.

Who Should Attend

Web services designers and programmers, application developers, and programmers using the Java programming language who have experience using the Java™ 2 Platform, Enterprise Edition (J2EE™).

Prerequisites

Experience using the Java programming language and basic knowledge of XML

Overview

This one-day seminar provides in-depth knowledge on Web services and shows how to develop Web services using the Java programming language and XML, the technologies of portable code and portable data respectively.

The session will start with an introduction on fundamental concepts and characteristics of Web services. This will be followed by a detailed explanation of how to implement, describe, register, discover, and invoke Web services using core Web services standards - Simple Object Access Protocol (SOAP); Web Services Description Language (WSDL); and Universal Description, Discovery, and Integration (UDDI). In addition, the ebXML standard, which defines the framework for the global electronic marketplace will be talked about in detail. Also, the tools for building and deploying Web services will be discussed. Each topic will be presented with concrete examples and demonstrations when possible.

Attendees will also learn how to use standard Java APIs for Web services, mainly Java API for XML Messaging (JAXM), Java technology API for XML-based RPC (JAX-RPC), and Java technology API for XML Registries (JAXR) for developing and deploying Web services.

Benefits

- Learn the fundamental concepts and characteristics of Web services. Gain detailed understanding on core Web services standards: SOAP, WSDL, UDDI.
- Gain a detailed understanding of ebXML, the standard framework for electronic business.
- Learn Java programming language APIs for Web services – JAXM, JAX-RPC, JAXR

Wednesday, March 19, 2003

Java™ 2 Platform: Programmer Certification Fast Path

PHILIP HELLER, PRESIDENT, HELLER ASSOCIATES

Who Should Attend

This session is designed for programmers who have some exposure to the Java™ programming language, and are ready to prepare for the Sun Certified Programmer for Java 2 Platform exam.

Prerequisites

Object-oriented software development experience and familiarity with the syntax and structure of Java technology-based development.

Overview

The development community recognizes that competency developing solutions using Java technology is vital to productivity, reaffirms your value to your organization, and increases your career advancement opportunities. This session, developed and delivered by Philip Heller, author of the two leading Java technology certification preparation manuals, helps to prepare you for the Sun Certified Programmer for the Java 2 Platform exam. Philip provides code-level, detailed review of the skills and knowledge needed to confidently approach the exam.

Benefits

- Receive an intensive review of the advanced topics covered on the Sun Certified Programmer for the Java 2 Platform Exam
- Increase your understanding and knowledge of Java programming language syntax and structure
- Prepare for the exam by reviewing practice tests and questions
- Gain a strong understanding of Java fundamentals

Thursday, March 20, 2003

Java™ 2 Platform: Architect Certification Fast Path

SIMON ROBERTS, TECHNOLOGY EXPERT AND COURSE DEVELOPER, SUN MICROSYSTEMS, INC.

Who Should Attend

This session is designed for enterprise application architects, system analysts, experienced technologists, and developers using Java™ technology seeking certification as an architect for the Java™ 2 Platform, Enterprise Edition (J2EE™).

Prerequisites

Understand the benefits of Java technology solutions; experience with object-oriented analysis and design; familiarity with concepts of distributed computing.

Overview

Many of the solutions in today's "Net economy" are, or soon will be, developed using the Java 2 Platform, Enterprise Edition (J2EE) architecture. Gaining recognized competency architecting J2EE platform-based solutions is vital to your success as an architect, reaffirms your value, and increases your career opportunities.

Developed and presented by Mark Cade, this intense one-day session helps prepare attendees to pass the Sun Certified Enterprise Architect for J2EE Technology exam. This session provides an overview of the components comprising the J2EE architecture as a whole, emphasizes the incorporation of J2EE technology into an architecture, and reviews each of the certification exam's testing objectives. Multiple real-world case studies are used to demonstrate correctly architected J2EE technology–based solutions and pinpoint key topics presented within the architect exam.

Additionally, you will learn how to interpret exam objectives, what each of the three exam phases contains, and clear guidelines and resources to use after the course.

Benefits

- Receive an intensive review of the topics covered on the Sun Certified Enterprise Architect for the Java 2 Platform, Enterprise Edition Exam
- Increase your understanding and knowledge of successfully architecting solutions using J2EE technology
- Understand the system qualities: scalability, availability, extensibility, performance, and security
- Understand trade-offs of different architectural choices as they pertain to system qualities.
- Describe the benefits and weaknesses of potential J2EE technology–based architectures
- State benefits and costs of persistence management strategies
- Review real-world case studies of J2EE technology–based architecture
- Prepare for the exam by reviewing practice tests and questions

Major Sponsors at the World's Largest Web Services, Java, XML, and .NET Conference and Expo

2003 East Gold Sponsors

COMPUWARE

Compuware will be the best worldwide provider of quality software products and services designed to increase productivity. We will continue to create practical solutions that meet our customers needs and surpass their expectations. We will provide an environment for our employees where excellence is encouraged and rewarded and where diversity is promoted at all levels of the company. Compuware is built upon a diverse employee population with a set of key attributes that is common to every individual: Secure, Moral, Ethical, In Agreement with Economic Purpose, Team Player, Other Oriented, Positive, Optimistic, Energetic Achiever and Bright.

Microsoft

Introducing Visual Studio .NET; visionary yet practical, the single comprehensive development tool for creating the next generation of applications has arrived. Developers can use Visual Studio .NET to:

- Build the next-generation Internet.
- Create powerful applications fast and effectively.
- Span any platform or device.

Visual Studio .NET is the only development environment built from the ground up for XML Web services. By allowing applications to share data over the Internet, XML Web services enable developers to assemble applications from new and existing code, regardless of platform, programming language, or object model.

ORACLE

Oracle Corporation (Nasdaq: ORCL) is the world's largest enterprise software company, providing enterprise software to the world's largest and most successful businesses. With annual revenues of more than $9.4 billion, the company offers its database, tools, and application products, along with related consulting, education, and support services. Headquartered in Redwood Shores, California, Oracle is the first software company to develop and deploy 100 percent Internet-enabled enterprise software across its entire product line: database, server, enterprise business applications, and application development, and decision support tools.

2003 East Silver Sponsor

hp invent

HP is a leading provider of products, technologies, solutions and services to consumers and business. The company's offerings span IT infrastructure, personal computing and access devices, global services, and imaging and printing. Our $4 billion (U.S.) annual R&D investment fuels the invention of products, solutions and new technologies, so that we can better serve customers and enter new markets. We invent, engineer, and deliver technology solutions that drive business value, create social value, and improve the lives of our customers.

2003 East Featured Sponsor

Sun microsystems

Sun was founded with one driving vision. A vision of computers that talk to each other no matter who built them. A vision in which technology works for you, not the other way around. While others protected proprietary, stand-alone architectures, we focused on taking companies into the network age, providing systems and software with the scalability and reliability needed to drive the electronic marketplace.

2003 East Corporate Sponsors

IONA

IONA is the leading e-Business Platform provider for Web Services Integration. IONA Orbix E2A enables the flow of information across disparate systems and the liberation of application assets for new business opportunities. For additional information about IONA, visit our Web site at http://www.iona.com.

Rational the e-development company™

Rational Software provides a software development platform that improves the speed, quality, and predictability of software projects. This integrated, full life-cycle solution combines software engineering best practices, market-leading tools, and professional services. Ninety-six of the Fortune 100 rely on Rational tools and services to build better software, faster. Additional information is available on the Internet at http://www.rational.com

SAMS

Sams Publishing has more than 500 titles in print, and is one of the most successful computer book publishers in the world. From introductory tutorials to comprehensive reference books, Sams Publishing focuses on teaching tomorrow's programmers, developers and system administrators the skills they need to build and manage emerging technologies.

SYBASE

Sybase has always delivered solutions that help customers to share data, is platform independent, and integrates everything: platforms, application servers, components, databases, portals, processes, message brokers, and mobile/wireless. Our technologies have the insight that will promote ease of use, leverage best practices, ensure positive ROI, and help your organization build a successful, pragmatic strategy based on next-generation technologies. Check us out at http://www.sybase.com/solutions/e-business

DIGITAL EVOLUTION The Business Value of Web Services

DE solutions and thought leadership deliver the value of Web services to business users today. DE unleashes the value of underlying IT assets, giving enterprises the freedom to innovate and evolve business processes within a standards-based, vendor neutral, loosely coupled framework. Some of the largest corporations in the world rely on DE's Web services solutions. www.digev.com

Special Insert: Web Services Edge East Conference & Expo

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

REGISTER ONLINE TODAY @ SYS-CON.COM

JDJ EDGE conference&expo | web services EDGE conference&expo | .NET EDGE conference&expo | XML EDGE conference&expo

Don't Miss the Developer Event of 2003

Meet with the industry experts, professionals, and visionaries connecting the enterprise with today s i-technology!

Conference
March 18–20, 2003

Exposition
March 19–20, 2003

Hynes Convention Center, Boston

WEB SERVICES EDGE CONFERENCE PAST SPONSORS & EXHIBITORS

XML Schemas in an Object-Oriented Framework

Established best practices and emerging design patterns

WRITTEN BY
AYESHA MALIK

XML Schemas provide the grammar and constraints of XML documents. Given the increasing prevalence of XML as a data and command transport format in enterprise software systems across industries, it's important to have a methodology when designing the structure of the XML you will create.

In this article, I will discuss how to leverage established object-oriented frameworks to design XML Schemas. The three tenets of object-oriented design – encapsulation, inheritance, and polymorphism – are examined along with emerging design patterns and when to apply them. Since industry consortia are creating industry XML standards, I will also discuss how to use and extend standards for your business needs. Finally, there is a brief overview of how to use Unified Modeling Language (UML) to follow the best practices described in this article.

Case Study: BigPharma

To illustrate, I'll use the fictional example of a large pharmaceutical company called BigPharma. BigPharma is one of the leading pharmaceutical companies in the world and is keen to take advantage of the genetic data rapidly becoming available. The retrieval and analysis of genetic data is done in the research department of the company. The research department must get the DNA sequences from different sources, analyze them using different analytical techniques, and then send the data to BigPharma's headquarters (see Figure 1). This analysis is then passed on to the drug-making department, and all this information is transported using XML. Since the data is complex and is crucial to the company's future profitability, BigPharma insists that the XML be structured properly and hires an expert to design the XML Schemas for the entire firm.

Encapsulation

The first thing the expert designs is a Data Types library for the firm. A Data Types library keeps all the common types used in a company in one place. For instance, the definition of an Employee is relevant to many departments in the firm – BigLab, Accounting, Drug Creation, Marketing – since all these departments contain and exchange information on employees. In order to create consistency in the definition used across the firm, the complex type Employee is kept in the Data Types library. You can name this file datatypes.xsd.

Whenever you need to send information about an employee, you will use the same definition of Employee. Observe the UML representation of Employee in Figure 2. You can see that Employee has a Name, an EmployeeID, and a Department. If BigLab's research.xml is using Employee, the schema research.xsd will just refer to Employee.

```
<xs:complexType name="Employee">
  <xs:sequence>
    <xs:element name="employeeID" type="xs:string"/>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="department" type="xs:string"/>
  </xs:sequence>
 </xs:complexType>
```

This is known as encapsulation because the XML document designer does not need to know and create the characteristics of Employee each time he or she uses it. In fact, Employee's information is encapsulated in the data type Employee, and just by referencing this, the document designer has access to the definition of Employee.

The way to access this data type is to include the datatypes.xsd library in your current schema. The included

schema takes on the target namespace of the enclosing schema like a chameleon, thereby earning the name *Chameleon Effect* (see Listing 1).

Inheritance

Inheritance is a key tenet of object-oriented design. It has many benefits, including object reuse and consistency. You don't want similar data types to be re-created each time and to be different in structure even though one is a slight extension of the other. In programming, inheritance is used through abstract classes or base types. In a similar vein, inheritance is possible in XML Schemas through extensions and restrictions.

Inheritance by extension

Continuing with our example, the research department sends research information in an XML document, Research.xml, to BigPharma's headquarters. Each research document contains analysis conducted on genetic data. Regardless of which kind of analytic algorithm is used, the analysis document must contain the name of the employee who conducted the analysis and the date of the analysis. In other words, the base data type, Analytics, must contain Employee and Date. By using the keyword *abstract="true"*, Analytics is converted into an abstract type. Any specific analytic algorithm can extend this type using the keyword *extension base="Analytics"* and then add its own specific metrics to its type (see Figure 3).

Say you're going to use the algorithm known as Dynamic Programming, which analyzes genetic data and gives a result known as "score". Create the complex type "DynamicProgramming" as an extension of Analytics. This will automatically ensure that anyone using the definition of DynamicProgramming in his or her XML will have to include Employee and Date information. In addition, you add the result "score" to your new type. The resulting XML instance document Research.xml will contain Score as well (see Listing 2).

Inheritance by restriction

Inheritance by restriction is a little unusual since it's not commonly used in programming languages. The idea is that you use the same data type but with some restrictions. For example, when a researcher is a consultant, he or she will still have all the characteristics of an Employee, i.e., name and department. However, since the consultant is not a permanent employee, the consultant does not have an employeeID. Using the keyword *restriction base = "Employee"*, we can create a new type known as Consultant that has employeeID missing from its composite elements (see Listing 3).

Blocking inheritance

XML Schemas provide a mechanism by which an XML architect can block extensions and derivations on data types that have been defined. In Java, developers declare an attribute final if they do not want anyone to tamper with it. One use case might be the definition of a consultant. BigPharma has very strict rules for what it considers an outside consultant. Given the sensitivity and private nature of the data required, it is imperative that consultant information be tagged with the description that headquarters provides. If BigPharma does not allow extensions, restrictions, or both, it uses the final keywords as shown below.

```
<xs:complexType name="Consultant" final="extension">
<xs:complexType name="Consultant" final="restriction">
<xs:complexType name="Consultant" final="#all">
```

Polymorphism

The word polymorphism stems from a Greek word meaning "different forms." In object-oriented programming, polymorphism is the ability to use different representations of an object in different contexts at runtime. Polymorphism is closely tied to the notion of inheritance: derived types have different representations of the base class characteristics.

In XML Schemas, polymorphism refers to the ability to have different types for the same component name. Regardless of which type you use, the XML instance document should validate because it is one of the forms of the component that is valid. For instance, our Analytics can be of several different types: Dynamic Programming, Hidden Markov Models, and Phylogenetic Trees. In your schema, you may specify

```
<xs:element name="analytics" type="Analytics"/>
```

but you can have any derived type of Analytics in your instance document and the XML will still validate against the schema. In the following code, you can see that Analytics is of type Dynamic Programming as it has the new element score in it which was added as an extension to the type Analytics. The important keyword to note in the instance document is the attribute *xsi:type*, which specifies the derived type being used. This is an example of polymorphic behavior.

```
<analytics xsi:type="DynamicProgramming">
  <employee>
   <employeeID>0988764</employeeID>
   <name>Albert Einstein</name>
   <department>Genetic Analysis</department>
  </employee>
  <date>1967-08-13</date>
```

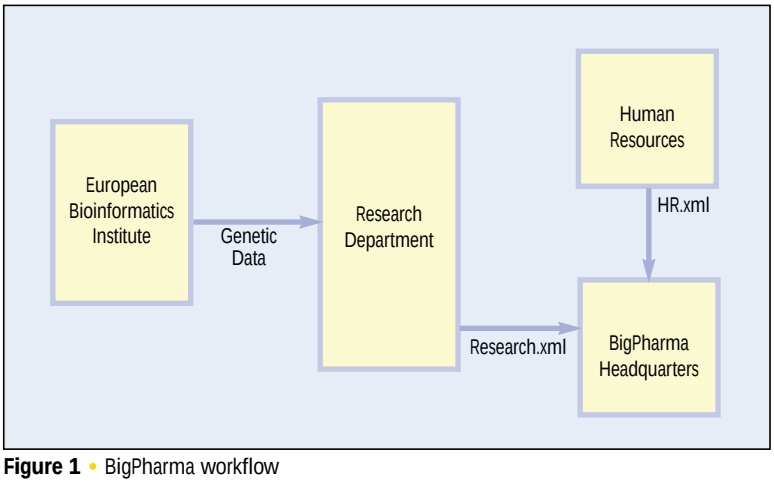


Figure 1 • BigPharma workflow

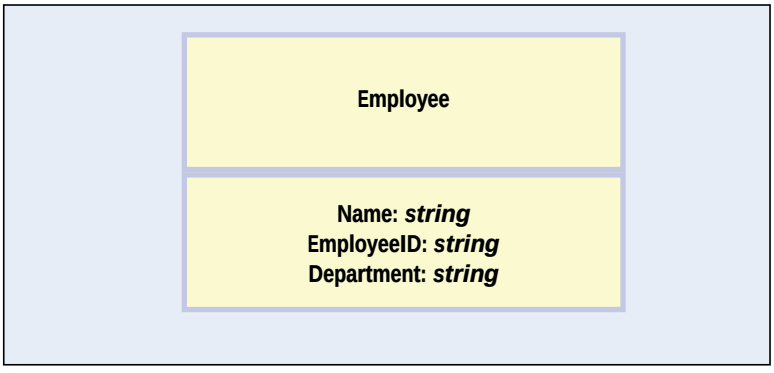


Figure 2 • UML representation of Employee data type

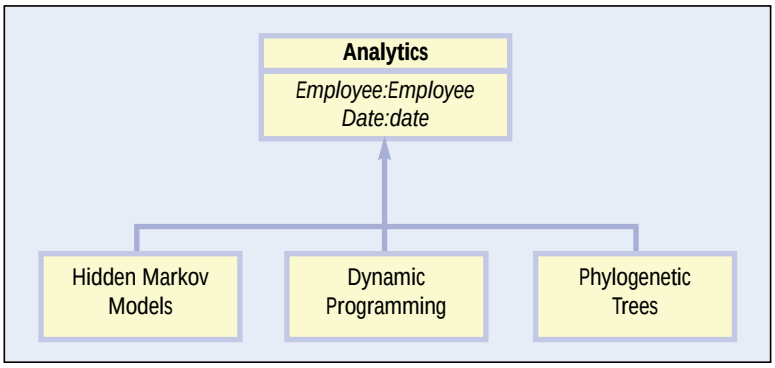


Figure 3 • UML diagram of Analytics abstract class with different extensions

```
    <score>3.141</score>
</analytics>
```

Emerging Design Patterns

Design patterns are established and tested methods of using a certain set of architectural solutions for specific business requirements. In the area of software programming, patterns have been well documented, for example, in the Gamma Book, and are used by developers as guidelines. Even though XML Schemas are a fairly recent phenomenon, some design patterns have been documented by experienced developers. Since XML Schemas represent static data, these patterns are about how to create data types and do not cover the behavioral requirements of a software system.

Currently, there are three main design patterns: Russian Doll, Salami Slice, and Venetian Blind.

Russian Doll

In the Russian Doll design pattern, all the internal types of a component are local to its definition. This means that types are nested within each other just as Russian dolls are nested inside one another. Russian Doll is so named because everything is a component within a component. Thus the Employee type contains the components employeeID, name, and department locally within its definition. If any other component wanted to access employeeID, it would be unable to achieve this (see Listing 4).

Russian Doll is a good pattern to use when schema size is a concern because it tightly couples components within each other.

Salami Slice

In the Salami Slice pattern, each element is declared a global component and is referenced when it is needed as a local element in a larger component. Salami Slice is so named because you put the types together like you put salami slices together to create a large salami sandwich. The keyword *ref* is used to access the global definitions of the component (see Listing 5).

While the Salami Slice pattern is more refactored (contains smaller units) than the Russian Doll pattern, it is unnecessary to create every component as a global element. It's usually recommended that elements specific to a particular component and not needed by other components be defined locally.

Venetian Blind

In the Venetian Blind pattern, every single component is created into a type. So in our example, EmployeeID will be a simpleType and will be accessed by the employeeID element used in Employee by *type= "EmployeeID"* (see Listing 6).

One advantage of this design pattern is that turning a component into a type means that it can have a namespace prefix. If you are using definitions from different namespaces, it may be necessary to identify the source of the definition (this is identified by the namespace prefix). Using the keywords elementFormDefault = "qualified" or "unqualified", you can either turn namespace prefixes on or off. This ability to either show or hide prefixes resembles the ability to open or close venetian blinds and is where the pattern gets its name.

Making Packages

Being able to separate function and service by units is an essential part of object-oriented design. In programming, developers create this separation by keeping classes in different packages. When designing XML Schemas, this can be achieved by keeping schemas in separate files and using different namespaces. For instance, generic services such as database functions are usually kept in packages such as dbservices, while accounting services are kept in an accounting package. With XML Schemas, you can keep the generic data types in a datatypes.xsd file. This file doesn't need to have a namespace and can be included in the different schemas. On the other hand, schemas for services such as research and human resources can be kept in research.xsd and hr.xsd. There are two advantages to this separation:

1. There is a clean logical separation of functions.
2. Different departments assume responsibility for their complex types and their definitions.

By keeping different namespaces, you always know the source of a definition (see Table 1).

Packages	Schemas
Research	Research.xsd
Human Resources	HR.xsd
Generic Services	DataTypes.xsd

Table 1 • Keeping separate namespaces

Integrating Industry XML Standards

More often than not, you'll find industry consortia defining data types that you need for internal use. It is always best to use an industry standard or even part of an industry standard definition when making your XML Schemas because then you pave the way for inter-firm communication. For instance, BigPharma may realize that it's sensible to use the definition of a genetic sequence as defined in the industry standard BSML (Bioinformatic Sequence Markup Language). While it's a good idea to use industry standards, how can BigPharma separate its internal definitions from the definitions defined by the standard.The easiest way to acheive integration of industry standards is to use namespaces by importing your extensions into the industry standard. By turning on namespace prefixes, you will always be able to separate internal additions from an industry standard. Namespace prefixes allow you to turn identification on or off just as described in the Venetian Blind design pattern discussed earlier.

UML for XML Schemas

Finally, there is always the question of how you can actually implement these modeling techniques. Every XML developer knows that writing XML Schemas is a difficult venture. Once again, we look to software architects to gain insight on how to visually model XML Schemas. In programming languages, it is now standard to use UML to create the models of the system. In

this article, I have used UML diagrams to show data types and inheritance in XML Schemas, and it's clear that there is good correspondence between class diagrams and type representation.

Problems may arise with UML because it doesn't capture all the richness of XML Schemas. For instance, representing XML Schema attributes in UML diagrams is difficult since there is no such concept in object-oriented programming. Fortunately, UML has a functionality by which you can create user-defined types through the use of UML Stereotypes. Rational Rose and Together J are two of the UML tools available that convert the UML diagrams into XML Schemas. However, it's better to manually check the generated schemas since they're not too stringent in the accuracy of the conversion. Even though automatic generation of schemas from UML has not been perfected in the industry until now, I highly recommend sketching your schema architecture using UML.

Summary

The design of XML Schemas is essential if you're going to use XML as a transport mechanism for your company's data. A good design will ensure consistency of definitions, maintainability across departments, and extensibility of and integration with external industry standards. Most of the best practices discussed in this article come from existing object-oriented practices, and emerging design patterns are created based on applying best practices to specific business requirements. Finally, when modeling XML Schemas, UML should be used as a modeling language both to communicate effectively with business analysts and to have a clear vision of how the schemas fit into the larger architectural design of the system.

As a note of caution, I urge you not to use the guidelines provided here to create very complex schemas. These guidelines are to be used to create simple XML Schemas that are architecturally flexible and extensible. Current technologies are ill equipped to handle and process schemas that are too complicated.

Resources

- *XFront's XML Schemas: Best Practices:* www.xfront.com/BestPracticesHomepage.html
- *Jeni Tennison Instructional Web site on XML Schemas:* www.jenitennison.com/schema/index.html
- *W3C Primer on XML Schemas:* www.w3.org/TR/xmlschema-0
- Malik, Ayesha. (October 2002). "Create Flexible and Extensible XML Schemas." IBM developerWorks: www-106.ibm.com/developerworks/xml/library/x-flexschema/?dwzone=xml
- Obasanjo, Dare. (July 2002). "W3C XML Schema Design Patterns: Dealing with Change." XML.com: www.xml.com/pub/a/2002/07/03/schema_design.html
- Carlson, Dave. (August 2001). "Modeling XML Schemas with UML." XML.com: www.xml.com/pub/a/2002/07/03/schema_design.html

Author Bio

Ayesha Malik is a senior software consultant at Object Machines and has worked extensively on large Java, XML, and Web services systems in a wide range of industrial environments. She is the author of articles on software development, and has been an invited speaker at several industry conferences. Ayesha holds a BA with honors from Harvard University and an MS from Columbia University, where she studied operations research, applied mathematics, and computer science.

AYESHA.MALIK@OBJECTMACHINES.COM

LISTING 1

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:include schemaLocation="datatypes.xsd"/>
 <xs:element name="research">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="employee" type="Employee"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
</xs:schema>
```

LISTING 2

```
<?xml version="1.0" encoding="UTF-8"?>
<research xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="research.xsd">
 <dynamicProgramming>
  <employee>
   <employeeID>0988764</employeeID>
   <name>Albert Einstein</name>
   <department>Genetic Analysis</department>
  </employee>
  <date>1967-08-13</date>
  <score>3.141</score>
 </dynamicProgramming>
</research>
```

LISTING 3

```
<xs:complexType name="Consultant">
  <xs:complexContent>
   <xs:restriction base="Employee">
    <xs:sequence>
     <xs:element name="name" type="xs:string"/>
     <xs:element name="department" type="xs:string"/>
    </xs:sequence>
   </xs:restriction>
  </xs:complexContent>
 </xs:complexType>
```

LISTING 4

```
<xs:complexType name="Employee">
  <xs:sequence>
   <xs:element name="employeeID" type="xs:string"/>
   <xs:element name="name" type="xs:string"/>
   <xs:element name="department" type="xs:string"/>
  </xs:sequence>
 </xs:complexType>
</xs:schema>
```

LISTING 5

```
<xs:element name="employeeID" type="xs:string"/>
 <xs:element name="name" type="xs:string"/>
 <xs:element name="department" type="xs:string"/>

      <xs:complexType name="Employee">
  <xs:sequence>
   <xs:element ref="employeeID"/>
   <xs:element ref="name"/>
   <xs:element ref="department"/>
  </xs:sequence>
     </xs:complexType>
```

LISTING 6

```
<xs:simpleType name="EmployeeID">
  <xs:restriction base="xs:string"/>
 </xs:simpleType>
 <xs:simpleType name="Name">
  <xs:restriction base="xs:string"/>
 </xs:simpleType>
 <xs:simpleType name="Department">
  <xs:restriction base="xs:string"/>
       </xs:simpleType>

 <xs:complexType name="Employee">
  <xs:sequence>
   <xs:element name="employeeID" type="EmployeeID"/>
   <xs:element name="name" type="Name"/>
   <xs:element name="department" type="Department"/>
  </xs:sequence>
     </xs:complexType>
```

PLE MAGAZINES ONLINE

AND SAVE UP TO $400 AND RECEIVE UP TO 3 FREE CDs!

RECEIVE YOUR DIGITAL EDITION ACCESS CODE INSTANTLY WITH YOUR PAID SUBSCRIPTIONS

TO ORDER

•Choose the Multi-Pack you want to order by checking next to it below. •Check the number of years you want to order. •Indicate your location by checking either U.S., Canada/Mexico or International. •Then choose which magazines you want to include with your Multi-Pack order.

Pick a 3-Pack, a 6-Pack or a 9-Pack						
☐ 3-Pack	☐ 1YR	☐ 2YR	☐ U.S.	☐ Can/Mex	☐ Intl.
☐ 6-Pack	☐ 1YR	☐ 2YR	☐ U.S.	☐ Can/Mex	☐ Intl.
☐ 9-Pack	☐ 1YR	☐ 2YR	☐ U.S.	☐ Can/Mex	☐ Intl.

☐ **Linux Business & Technology**

U.S. - Two Years (24) Cover: $143	You Pay: $79.99 /	Save: $63 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $39.99 /	Save: $32
Can/Mex - Two Years (24) $168	You Pay: $119.99 /	Save: $48 + FREE $198 CD
Can/Mex - One Year (12) $84	You Pay: $79.99 /	Save: $4
Int'l - Two Years (24) $216	You Pay: $176 /	Save: $40 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **Java Developer's Journal**

U.S. - Two Years (24) Cover: $144	You Pay: $89 /	Save: $55 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $49.99 /	Save: $22
Can/Mex - Two Years (24) $168	You Pay: $119.99 /	Save: $48 + FREE $198 CD
Can/Mex - One Year (12) $84	You Pay: $79.99 /	Save: $4
Int'l - Two Years (24) $216	You Pay: $176 /	Save: $40 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **Web Services Journal**

U.S.- Two Years (24) Cover: $168	You Pay: $99.99 /	Save: $68 + FREE $198 CD
U.S. - One Year (12) Cover: $84	You Pay: $69.99 /	Save: $14
Can/Mex - Two Years (24) $192	You Pay: $129 /	Save: $63 + FREE $198 CD
Can/Mex - One Year (12) $96	You Pay: $89.99 /	Save: $6
Int'l - Two Years (24) $216	You Pay: $170 /	Save: $46 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **.NET Developer's Journal**

U.S. - Two Years (24) Cover: $168	You Pay: $99.99 /	Save: $68 + FREE $198 CD
U.S. - One Year (12) Cover: $84	You Pay: $69.99 /	Save: $14
Can/Mex - Two Years (24) $192	You Pay: $129 /	Save: $63 + FREE $198 CD
Can/Mex - One Year (12) $96	You Pay: $89.99 /	Save: $6
Int'l - Two Years (24) $216	You Pay: $170 /	Save: $46 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **XML-Journal**

U.S. - Two Years (24) Cover: $168	You Pay: $99.99 /	Save: $68 + FREE $198 CD
U.S. - One Year (12) Cover: $84	You Pay: $69.99 /	Save: $14
Can/Mex - Two Years (24) $192	You Pay: $129 /	Save: $63 + FREE $198 CD
Can/Mex - One Year (12) $96	You Pay: $89.99 /	Save: $6
Int'l - Two Years (24) $216	You Pay: $170 /	Save: $46 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **WebLogic Developer's Journal**

U.S. - Two Years (24) Cover: $360	You Pay: $169.99 /	Save: $190 + FREE $198 CD
U.S. - One Year (12) Cover: $180	You Pay: $149 /	Save: $31
Can/Mex - Two Years (24) $360	You Pay: $179.99 /	Save: $180 + FREE $198 CD
Can/Mex - One Year (12) $180	You Pay: $169 /	Save: $11
Int'l - Two Years (24) $360	You Pay: $189.99 /	Save: $170 + FREE $198 CD
Int'l - One Year (12) $180	You Pay: $179 /	Save: $1

☐ **ColdFusion Developer's Journal**

U.S. - Two Years (24) Cover: $216	You Pay: $129 /	Save: $87 + FREE $198 CD
U.S. - One Year (12) Cover: $108	You Pay: $89.99 /	Save: $18
Can/Mex - Two Years (24) $240	You Pay: $159.99 /	Save: $80 + FREE $198 CD
Can/Mex - One Year (12) $120	You Pay: $99.99 /	Save: $20
Int'l - Two Years (24) $264	You Pay: $189 /	Save: $75 + FREE $198 CD
Int'l - One Year (12) $132	You Pay: $129.99 /	Save: $2

☐ **Wireless Business & Technology**

U.S. - Two Years (24) Cover: $144	You Pay: $89 /	Save: $55 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $49.99 /	Save: $22
Can/Mex - Two Years (24) $192	You Pay: $139 /	Save: $53 + FREE $198 CD
Can/Mex - One Year (12) $96	You Pay: $79.99 /	Save: $16
Int'l - Two Years (24) $216	You Pay: $170 /	Save: $46 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **WebSphere Developer's Journal**

U.S. - Two Years (24) Cover: $360	You Pay: $169.99 /	Save: $190 + FREE $198 CD
U.S. - One Year (12) Cover: $180	You Pay: $149 /	Save: $31
Can/Mex - Two Years (24) $360	You Pay: $179.99 /	Save: $180 + FREE $198 CD
Can/Mex - One Year (12) $180	You Pay: $169 /	Save: $11
Int'l - Two Years (24) $360	You Pay: $189.99 /	Save: $170 + FREE $198 CD
Int'l - One Year (12) $180	You Pay: $179 /	Save: $1

☐ **PowerBuilder Developer's Journal**

U.S. - Two Years (24) Cover: $360	You Pay: $169.99 /	Save: $190 + FREE $198 CD
U.S. - One Year (12) Cover: $180	You Pay: $149 /	Save: $31
Can/Mex - Two Years (24) $360	You Pay: $179.99 /	Save: $180 + FREE $198 CD
Can/Mex - One Year (12) $180	You Pay: $169 /	Save: $11
Int'l - Two Years (24) $360	You Pay: $189.99 /	Save: $170 + FREE $198 CD
Int'l - One Year (12) $180	You Pay: $179 /	Save: $1

3-Pack

Pick any 3 of our magazines and save up to $275.00

Pay only $175 for a 1 year subscription plus a FREE CD

- 2 Year – $299.00
- Canada/Mexico – $245.00
- International – $315.00

6-Pack

Pick any 6 of our magazines and save up to $350.00

Pay only $395 for a 1 year subscription plus 2 FREE CDs

- 2 Year – $669.00
- Canada/Mexico – $555.00
- International – $710.00

9-Pack

Pick 9 of our magazines and save up to $400.00

Pay only $495 for a 1 year subscription plus 3 FREE CDs

- 2 Year – $839.00
- Canada/Mexico – $695.00
- International – $890.00

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

An Introduction to BSML

WRITTEN BY
KRISTIAN CIBULSKIS

Enabling the bioinformatics revolution

Over the past two decades, the life sciences industry has taken a dramatic leap into an online, collaborative world. Tasks and activities that are commonplace today were either extremely difficult or outright impossible just 20 years ago.

One key reason for this shift was the abundance of genomic sequence data, the sequence of base pairs that make up an organism's DNA. A vast amount of this data was made possible by advances in genomic sequencing techniques, such as the shotgun and clone contig techniques made famous to the general public during the Human Genome Project. In order to provide access to all of this publicly available data, the National Institute of Health created GenBank, an online repository of genomic information. When GenBank first opened in 1982 there were just 606 sequences available. Today, there are nearly 15 million sequences available – the number has doubled nearly every year since 1992!

With this vast amount of publicly available information came the need to share, organize, and analyze it. Bioinformatics was the term coined to describe this science. While bioinformatics is not an extremely well-defined term, in general it refers to three broad areas: the collection of new genomic data, the analysis and interpretation of existing data, and the development of new algorithms for analysis.

While the Internet has always been an important tool for researchers, it was a key catalyst for the bioinformatics revolution. Through the Internet, scientists were able to publish and analyze data in a truly collaborative fashion. One group of researchers could analyze a specific genome in a particular way to gain insight as to its function. They could then describe their new understanding of the genome sequence by annotating it and sharing that annotation. Research groups could then leverage the ongoing work of other groups through public data repositories. However, as with many areas on the Internet, a common format was required in order to efficiently share information.

Entrez XML

Early on, flat file formats were the de facto standard for information exchange. However, the larger the variety of information, the more difficult it became to capture it all. No longer was information sharing limited to the familiar raw genomic strings of A, T, C, and G. Broader annotations, research protocols, published research results, metabolic pathways, and more now needed to be shared and cross-referenced.

From its inception GenBank standardized on a format known as ASN.1. In fact, the ASN.1 format is still used by several large government organizations, including GenBank, as a default format. It is an object-oriented, text-based language that in many ways resembles XML. However, with the advent of XML it became apparent that XML would become the backbone of a future standard. Not only could XML represent arbitrarily complex information, as was required, but it was also eagerly adopted by the general software development community. This made it much easier to write new tools that bioinformatics scientists relied upon to process genomic data.

Of course, XML is an excellent choice for data representation in many industries. However, in order for the value of XML to be fully realized, the majority of producers and consumers in a given industry must use a common schema. Early on, several XML formats were created and backed solely by commercial companies. Each vendor spoke their own particular flavor of XML, and so a common format was not achieved.

Finally, in 1997, one standard emerged as the result of a grant from the National Human Genome Research Institute (NHGRI). Its charter was to create a public domain standard to communicate genomic research information.

BSML

The standard that emerged from the NHGRI grant was an XML-based language known as the Bioinformatic Sequence Markup Language, commonly referred to as BSML. It was developed and revised by LabBook scientists: Eluem Blyden, Dean Dai, David Gordon, Chaobo Guo, Seth Kraut, Eric Rentschler, Steven Roggenkamp, Robert Rumpf, Jeff Spitzner, and Joe Spitzner. LabBook is a key contributor to BSML and provides value-add products built around the standard.

From the beginning, BSML was the property of the public domain. There are no licensing agreements or fees required to use BSML, as is the case with most successful XML standards. All of the reference material for the standard, including the DTD and a reference guide, is maintained at www.bsml.org.

BSML seeks to encode three distinct types of information:

1. ***Definitions:*** Biological molecules, such as DNA, RNA, and protein sequences. In addition to the raw sequence data it is also possible to store sequence annotations, also known as features, and results of performed analysis.
2. ***Research:*** Queries, analyses, and experiment protocols. This research information can be cross-referenced with the definition of the molecule being studied.
3. ***Display:*** Graphical metaphors that can be used to visualize the above biological information. These metaphors are described as primitive widgets that are nonspecific to a platform or technology.

The Definitions aspect of BSML allows for the expression of

what is being studied, the particular DNA or protein sequence, for example. Building upon that data, the Research elements can describe how something is being studied, such as the protocol used for a particular experiment. Finally, the Display section provides a mechanism for representing the "what" and "how" data in a specific, meaningful manner. While the Definitions aspect of BSML is extremely valuable, it is this encoding of queries, visualizations, and cross-references that causes BSML to progress higher up the XML value chain, as shown in Figure 1.

Industry Support

In order for any XML format to become an industry standard, it must be endorsed by the standards bodies and leading companies of that industry. BSML is in an excellent position to receive this kind of support.

In addition to the support of NHGRI, the Interoperable Informatics Infrastructure Consortium (I3C) also endorses BSML. The I3C is a collection of leading companies seeking to create open standards for the bioinformatics industry. Also, the BSML standard has recently been submitted to the American Society for Testing and Materials (ASTM). The ASTM has a close relationship with ANSI, which should lead to a rapid endorsement from ANSI as well.

Furthermore, the BSML standard is actively supported by some of the biggest names in the life sciences industry, such as Bristol-Myers Squibb, NetGenics, and IBM, as well as a number of open-source communities such as BioPerl.

BSML Examples

But enough about the history, features, and support of BSML – let's take a look at several examples of BSML encoded data. One of the most basic types of information that BSML can encode is biological molecule sequence information. For example, take the BSML fragment shown in Listing 1.

The <bsml> tag is the outermost tag used to describe a BSML document. Within that tag, there are three major subtags: <definitions>, <research>, and <display>, which correspond to the three major types of information the BSML is intended to represent. In this case, sequence data is being defined. Other tags are available to encode other biological molecule information, such as genomes, isoforms, and networks. Table 1 breaks down the <sequence> tag.

As research proceeds on a given biological molecule, certain segments of the sequence become interesting for a variety of reasons. Sequence annotation is used to capture this extra information about the sequence data. Positional annotation refers to annotations that are specific to a portion of a sequence. In BSML, positional annotation is captured through Feature tags. Feature tags are child tags of a sequence tag, and therefore a Feature is related to a single sequence. For example, the following tag indicates that the region between 1513 and 1962 encodes a particular gene:

```
<Feature id="FTR4" title="Leucine TNRA" class="GENE">
  <Qualifier value-type="gene"/>
  <Interval-loc startpos="1513 endpos="1962"
    complement="0"/>
</Feature>
```

A given DNA sequence could have many features associated with it. Rather than simply encoding all of these flatly, in BSML related feature tags can be aggregated into Feature-Tables. Feature-Tables are intended to provide a logical grouping to features, such as grouping all gene expression features together.

An annotation can also take the form of a comparison between two sequences. Perhaps two segments are equivalent to one another. In order to achieve this in BSML, a <segment-set> tag can be used to enclose a set of segments represented by <segment> tags. For example, the tag shown in Listing 2 expresses that a region from sequence AB1432 and sequence NZ5723 are equivalent.

The type attribute on the <segment-set> element is used to indicate the relationship between the segments in the segment-set. Table 2 contains a list of the other possible relationships.

In addition to capturing base Definition data, BSML can also express the research used to obtain that data. For example, significant data can be obtained by executing a query against online genomic repositories. While the results of such a query can be stored in the Definition region of a BSML document, BSML also allows for the capture and reference of that result with the query that produced it. In this way, other researchers can easily duplicate and build upon previous research. The research fragment shown in Listing 3 captures the parameters used in a GenBank search.

Ideally, research tags should include enough information for another researcher to duplicate the research. In the above example, the name of the public database along with a URL and the parameters used to execute the search are captured.

While BSML is able to capture the base definition data and research used to obtain it, raw XML isn't necessarily the easiest

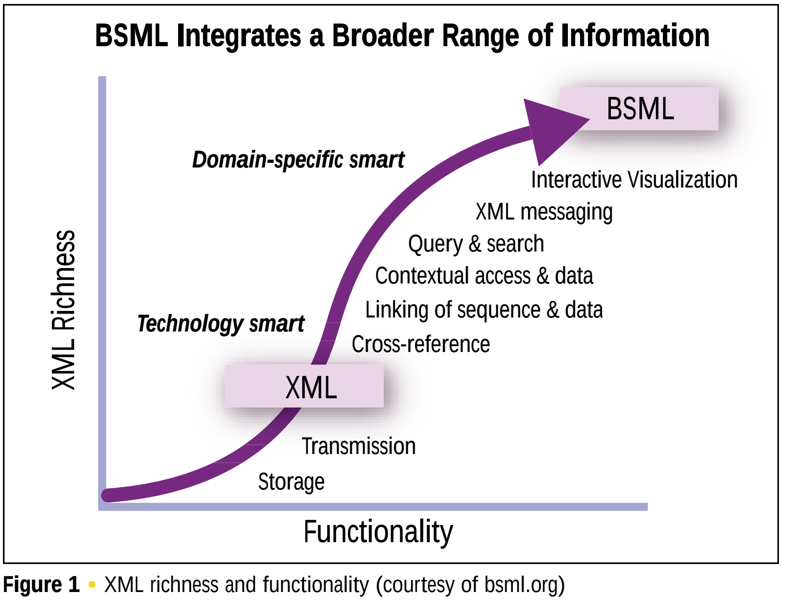


Figure 1 • XML richness and functionality (courtesy of bsml.org)

Attribute	Definition
id	A unique identifier
title	The displayable name of this sequence
molecule	The type of molecule being encoded, in this case DNA. Other possible values include rna, aa (Amino Acid)
length	Number of base pairs in this sequence
topology	Shape of this DNA fragment, typically linear or circular
strand	Indicates if this sequence of DNA is either single (ss) or double (ds) stranded
representation	Indicates how the <seq-data> tag encodes the DNA fragment. In this case the raw base pairs (A,C,T,G) are used. In addition to directly encoding the data, a reference to external sources can also be used.

Table 1 • Sequence example data

Type	Meaning
segment	Defines a single region of interest on a sequence
copy	One sequence was obtained through a copy of another sequence
translated	One sequence is a translated region of the other, usually the two sides of the same double strand of DNA
expressed	One sequence is the cDNA derived from the other DNA sequence
gapped	One sequence has base pair insertions, but is semantically equivalent to the other
aligned	The sequences are aligned regions, which means they match according to a scoring algorithm that takes mutations, etc., into account.

Table 2 • Segment set types

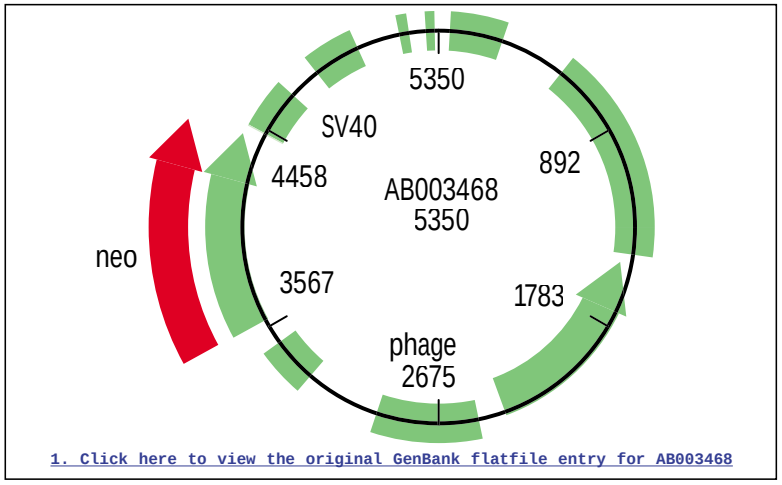


Figure 2 • Rendering of BSML display widgets

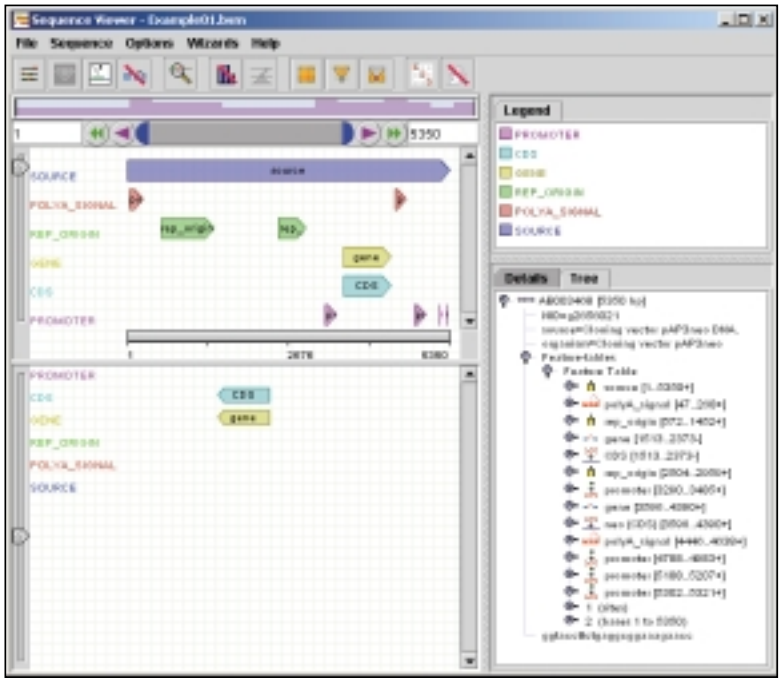


Figure 3 • Genomic XML Viewer sequence display

format for humans to digest. Even though graphical tree-like XML viewers are helpful, oftentimes a graphical metaphor for biological data can be invaluable. In BSML, these graphical constructs are called display widgets.

A few of the widgets in BSML are nonbiological such as a <caption-widget> or a <symbol-key-widget>, which represent a caption and a symbol legend, respectively. However, most of the display widgets are tightly coupled with biological concepts. The <Interval-widget> element is a graphical representation of an interval-based feature on a sequence. Extremely complex widgets, such as one that represents the rendering of electrophoresis gel images, are also available. Any BSML-aware application could render these widgets into a high-level UI for the user. For example, the XML fragment shown in Listing 4 is rendered by the Genomic XML Viewer as shown in Figure 2.

Because the display widgets are referenced to the underlying features through the featureref attributes, mouse-over and drill-down features allow the user to quickly and intuitively explore their data.

Genomic XML Viewer

BSML is commonly used as input to analysis programs that crunch upon the encoded data. However, one specific consumer of BSML is a generic viewer that makes it easy for a human to navigate and interact with the content of a BSML document. LabBook provides such a tool in their free Java-based Genomic XML Viewer, which not only interprets and displays BSML documents, but also is able to convert from other formats into BSML.

Figure 3 contains a screenshot of the Sequence Viewer portion of the Genomic XML Viewer. Graphical representations of all the features that have been defined on this sequence are shown in the left pane. The right pane contains a legend along with a view of the source XML tree where the sequence and feature data can been seen graphically. Using the Genomic XML Viewer is certainly more intuitive than using raw BSML.

In addition to the free Genomic XML Viewer, LabBook also promotes a commercial version called the Genomic Browser. While the Genomic XML Viewer provides a way to read BSML documents, the Browser allows for full creation, editing, and analysis of BSML documents.

Information Exchange

As with any emerging standard, there are always other competing and complementary standards. In the case of BSML, two other bioinformatics standards exist in a similar space: AGAVE and ASN.1.

The Architecture for Genomic Annotation, Visualization and Exchange (AGAVE) is an XML-based public standard originally developed at DoubleTwist, Inc., for its customers and partners. Unfortunately DoubleTwist closed down in March of 2002, and since then the AGAVE standard has been promoted by one of the original developers. It is still being considered by the I3C as a potential bioinformatics standard, and a number of customers are still using it. Comparatively, BSML aims to provide a broader scope of representation than AGAVE, such as research and improved sequence annotation features. A detailed comparison of the two standards can be found on the BSML Web site.

ASN.1, formally known as Abstract Syntax Notation number 1, is an object-oriented, hierarchical, text-based format for transmitting data between systems. It predates XML but provides many of the same features. ASN.1 is not specifically designed for life sciences but rather, like XML, can be used to represent any kind of structured data. Any type of data can be represented in ASN.1 once a schema is designed for it. ASN.1 is the format that GenBank initially used to publish genomic information and is still used to this day. The original goal of BSML was to be able to expand upon the set of information that ASN.1 could represent with respect to genomics and at the same time move to the more widespread technology of

XML, which is natively supported by every major programming language. In that respect BSML could even be considered an evolution of ASN.1 in the life sciences arena.

One of the core strengths of BSML, however, is the availability of public converters to translate from other formats into BSML. This allows consumers of bioinformatics data to pull together information from disparate sources into a single common language for their research. Surprisingly enough, many of these converters were not developed by LabBook, the company driving BSML as a standard, but rather from third-party adopters and supporters of BSML. For example, Bristol-Myers Squibb has release an open-source adapter into the BioPerl project that translates between the SeqIO format and BSML. Similarly, Cold Spring Harbor Laboratory has released a translator between the ASN.1 format used by GenBank and BSML. The European Bioinformatics Institute provides a translation between EMBL documents and BSML. Every day more and more translators become available, making it possible for researchers and application developers to build tools around BSML while accessing a variety of data sources.

Adopters and Supporters

There are two major types of BSML users in the bioinformatics world: producers who generate new BSML documents, and the consumers who use those documents. In general the largest producers of BSML are the public genomic databases as well as the translation programs that translate from another format into BSML. While most of the online repositories are still supporting their legacy data formats, many third parties are providing translation layers. These translation layers are usually Web based and provide a similar interface to the underlying data.

Internally, many life science companies are standardizing on BSML for their applications. It is able to capture the variety of data that needs to be handled, and allows for easier integration between systems, even within a single organization. The BSML Web site contains a list of companies who support and are using the BSML standard. However, at the same time, many companies in the life science space are not inclined to discuss the architecture of their confidential and proprietary software, so the true number of BSML adopters may be even higher.

Acknowledgement and Further Information

I'd like to thank Dr. Shawn Green of LabBook for his invaluable insight and feedback while researching this article. I also highly recommend the following Web sites for anyone interested in learning more about BSML:

- *BSML Home Page:* www.bsml.org
- *LabBook:* www.labbook.com (Genomic XML Viewer)
- *National Center for Biotechnology Information (GenBank):* www.ncbi.nlm.nih.gov
- *I3C:* www.i3c.org
- *European Bioinformatics Institute (EBML-BSML):* www.ebi.ac.uk/xembl

Author Bio

Kristian Cibulskis is freelance consultant focusing on high-performance J2EE and Oracle solutions for the life sciences industry. He is also the coauthor of the newly released ColdFusionMX: J2EE Integration *from Macromedia Press.*

KCIBUL@HOTMAIL.COM

LISTING 1

```
<bsml>
  <definitions>
    <sequences>
      <sequence id="AB12345" title="AB12" molecule="dna"
                length="500" topology="linear"
                  strand="ds"
                representation="raw">
        <seq-data>acgtacgtacgtacgtacgtcgcgaacgccg
          taact...</seq-data>
      </sequence>
    </sequences>
  </definitions>
</bsml>
```

LISTING 2

```
<segment-set type="equivalent">
  &lt;segment seg-source-type="source" seg-id="AB1432"
           seg-start="100" seg-end="300" seg-on-
            complement="0"/>

  &lt;segment seg-source-type="source" seg-id="NZ5723"
           seg-start="762" seg-end="962" seg-on-
            complement="0"/>
</segment-set>
```

LISTING 3

```
<Research>
  <Searches>
    <Search id="SCH1" database="GenBank" url="http://…">
      <Search-conditions>
        keyword=topoisomerase
      </Search-conditions>
    </Search>
  </Searches>
</Research>
```

LISTING 4

```
<Interval-widget class="GENE" title="gene" startpos="3596"
                     endpos="4390" featureref="FTR8"
                      auto-offset="1"
                     end-type="arrow" on-sequence="plus">
    </Interval-widget>
    <Interval-widget class="CDS" title="CDS"
      startpos="3596"
                     endpos="4390" featureref="FTR9"
                      auto-offset="1"
                     end-type="arrow" on-sequence="plus"
                     plus-text="neo" fill-fg-color="
                      #-cccd">
    </Interval-widget>
    <Interval-widget class="POLYA_SIGNAL" title="polyA_
      signal"
                     startpos="4446" endpos="4638"
                      featureref="FTR10"
                     auto-offset="1" on-sequence=
                      "center">
    </Interval-widget>

 …other interval widgets omitted…

    <Caption-widget text="1. Click here to view the
      original GenBank flat file entry for AB003468"
      font="FNT2" onclick="LNK2">
      <Coord hloc="1.1in" vloc="4.821in"/>
      <Link title="GenBank entry" id="LNK2"
            href="pap3neo.gbk" show="new" actuate="user"/>
    </Caption-widget>>
```

Download the Code
www.sys-con.com/xml

WRITTEN BY **AYESHA MALIK**

XBRL for Business Reporting

Positioned to become the global standard

Industry consortia are increasingly coming together to draw up XML standards for businesses.

One of the major emerging standards is XBRL: eXtensible Business Reporting Language. Organized by more than 170 finance, accounting, regulatory, and software companies from across the world, XBRL is emerging as the definitive standard for expressing business information contained in corporate documents. With XBRL, each piece of a company's reported information is represented in XML and can be searched and extracted depending on business requirements. Given that more than 80% of major U.S. public companies provide some type of financial disclosure on the Internet, XBRL is invaluable for stakeholders, analysts, investors, and regulators because it streamlines the collection and electronic distribution of this information.

The XBRL standard is an open, freely licensed standard and can be downloaded as an XML Schema at the XBRL.org Web site. In March 2002, Microsoft became the first technology company to publish its financial statements on the Internet using XBRL. Since then, companies such as Morgan Stanley, EDGAR Online, Reuters, and DaimlerChrysler have started to use it to document their business reports. XBRL is also used in several countries outside the U.S.: every lending institution in Australia, for example, currently reports to the Australian Prudential Regulatory Authority using XBRL. Already, vendors like Microsoft, SAP, Oracle, and Hyperion have committed to XBRL-enabling their financial-reporting applications.

AUTHOR BIO

Ayesha Malik is a senior software consultant at Object Machines and has worked extensively on large Java, XML, and Web services systems in a wide range of industrial environments. She is the author of articles on software development, and has spoken at several industry conferences. Ayesha holds a BA with honors from Harvard University and an MS from Columbia University, where she studied operations research, applied mathematics, and computer science.

The Business Requirement

Companies create many kinds of business reports, including financial statements, general ledger transactions, and regulatory filings for various business requirements, including credit risk analysis, consolidations, and tax filings. Some of these reports have to be created and published at regular intervals, such as annual and quarterly financial statements. To add to the complexity of financial reporting, the company must conform to the reporting standards of the jurisdiction in which it is located and must be prepared to append the reports when new laws are legislated. Formats can vary from Web pages to paper documents to PDF files and are transported via the Internet, mail, faxes, and e-mail attachments. Often, the information is reentered at several stages, which leads to errors in reporting and is overall an inefficient manner to transport information between disparate systems.

XBRL brings the advantages that XML and XML Schemas have shown in meeting business requirements of flexible, extensible, and standard transport formats. It allows for extensions and variation according to country-specific financial regulations, and its standard tags allow for consistency in reporting error-free documents in various formats. Since the information is in XML, it can be published to the Web, or converted by XSL transformations and built into PDF documents. In addition, XBRL allows for greater interoperability and speed of data exchange between disparate systems. All these advantages lead to reporting transparency of companies, and more powerful analysis of financials, and therefore, greater efficiency in the global capital markets.

XBRL is also invaluable in saving time for its users. "The way information is formatted today, in paper and electronic-paper form, it would take five or six hours to get information on, say, the five largest semiconductor companies and put it into an analytical application of any kind," says Mike Willis, founding chairman of XBRL International and partner at PricewaterhouseCoopers. "With the pilot program using XBRL, you could do it in 30 seconds."

The key to the XBRL technical specification consists of a framework composed primarily of taxonomies. The taxonomy describes a group of concepts for a particular financial area. For example, an electronic annual report can contain financial statements, auditor's reports, and notes, all coded and identified in the XBRL structure and tags.

XBRL Framework

XBRL focuses specifically on business reporting covering arenas such as U.S. GAAP (Generally Accepted Accounting Principles), IAS (International Accounting Standards), and Japanese GAAP. It is built on three XML specifications: XML Schema, XML Namespaces, and XML Linking. The XBRL Framework consists of two main concepts: items and taxonomies.

Items

The most fundamental concept in the framework is an item. An item corresponds to a fact such as a numeric value representing annual revenues for a particular year or text describing the principles used to generate that value. The specific time period and specific business entity must accompany the fact. In the XBRL schema, item is an abstract element. All the elements that are defined in taxonomies can substitute for item.

```
<ci:capitalLeasedAssetsNet.capital-
LeasedAssetsGross
numericContext="c1">727
</ci:capitalLeasedAssetsNet.capital-
LeasedAssetsGross>
```

The namespace prefix ci indicates that the number is calculated according to the GAAP Commercial & Industrial (ci) classification. A numerical fact, such as 727 shown in the code above, can only be understood with a numericContext.

A numericContext points to a definition that includes other elements such as :

An "entity/identifier" … IBM
A "period/instant" … 2002-02-11
A "unit/measure" … ISO4217:USD

Tuples

A tuple, like a row in a database table, is a grouping of facts. For instance, the name, age, and compensation of a director of a company should be grouped together to be correctly understood.

Groups

In XBRL, a group is a set of related items that can appear in any order and can be interspersed among other text and elements in any XML document. The root of an XBRL Instance Document is a group, which may, in turn, contain other groups. There is therefore no "XBRL document type" as such. It is possible in principle to embed an XBRL item in any document, such as a press release that is otherwise formatted in HTML.

Taxonomies

A taxonomy corresponds to a group of well-defined concepts such as the U.S. GAAP applied to Commercial and Industrial (CI) companies. Elements in a taxonomy correspond to the notion of elements in XML Schemas. An element in the taxonomy mentioned can be "Accounts Receivable Trade, Gross".

Taxonomies can be grouped together to represent larger taxonomies and can be extended to include new concepts. Essentially, the creation and extension of taxonomies follow the architectural rules of XML Schemas. XBRL considers the ability to accommodate virtually any business entity's unique reporting requirements through extensions as critical to its universal acceptance. An instance document, therefore, represents a collection of financial facts using tags from one or more taxonomies.

The CI taxonomy describes the creation of business reporting objects according to the GAAP Commercial & Industrial classification, whereas the IAS taxonomy does so for International Accounting Standards. In XBRL instance documents, the different classification and standards applied to a concept are indicated by the namespace prefix of the item (see Figure 1).

According to XBRL, "Taxonomies represent up to hundreds of individual business reporting concepts, mathematical and definitional relationships among them, along with text labels in multiple languages, references to authoritative literature, and information about how to display each concept to a user."

Period

A period is an instant or duration of time. In business reporting, financial numbers and other facts are reported "as of" an instant or for a period of certain duration. Items that report on instants and durations are both common.

Namespaces

Proper detection of an XBRL document depends on the XBRL namespace. The namespace for the current version is http://www.xbrl.org/2001/instance.

Linkbases

To represent the definition of relations between taxonomies, a taxonomy also uses a series of elements that implement a set of Xlink-compliant linkbases. XLink is a W3C standard way to use XML to represent all kinds of relationships between XML elements no matter where they are. Examples of linkbases include relation links (calculation, definition, and presentation) that manage the relations between taxonomy elements, and label links that manage the text associated with taxonomy elements in various languages.

Figure 2 shows some sample XBRL content.

Summary

XBRL is well positioned to become the global standard for business reporting. Not only is the standard already in use by leading companies in the U.S., it is also becoming popular amongst firms and stock exchanges in other countries. The most recent example of XBRL's adoption is the Toronto Stock Exchange in Canada. In the U.S., The American Institute of Certified Public Accountants (AICPA) is one of the organizations leading the XBRL effort.

Currently, XBRL is primarily used to digitally publish financial statements of companies that are issued to external users. An XBRL-based financial statement is a digitally enhanced version of paper-based financial statements, which include the balance sheet, income statement, statement of equity, statement of cash flows, and the notes to the financial statements as well as the accountant's report. Using XBRL greatly increases the transparency, speed, and accuracy of financial statements for investors. Since XBRL is XML schema-based, you can extend its taxonomies to include your unique business reporting needs if XBRL's current coding falls short of your requirements.

AYESHA.MALIK@OBJECTMACHINES.COM

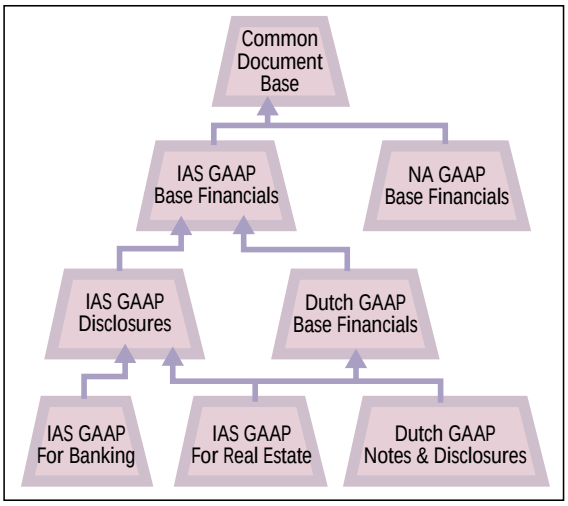


Figure 1 • Extending taxonomies (Source: "XBRL for XML Savvy" by Walter Hamscher)

```
<numericContext id="rg.cy00.hkd" cwa="false" precision="4">
  <entity>
    <identifier scheme="http://www.gov.hk">rg</identifier>
  </entity>
  <period>
  <startDate>2000-01-01</startDate>
  <endDate>2000-12-31</endDate>
  </period>
  <unit>
    <measure>iso4217:hkd</measure>
  </unit>
</numericContext>
```

Rock Gravel Corporation
Period from 1 Jan 2000 to 31 Dec 2000
In HKD
Precise to 4 digits

Figure 2 • Sample XBRL content (Source: "XBRL for XML Savvy" by Walter Hamscher)

Microsoft BizTalk Server

Hitesh Seth, editor-in-chief of XML-Journal and XML Track chair for Web Services Edge, is the chief technology officer of ikigo, Inc., a provider of Business Activity Monitoring solutions.

REVIEWED BY **HITESH SETH**

Microsoft BizTalk Server 2002

Part 1

An out-of-box integration solution

Microsoft Corporation

One Microsoft Way
Redmond, WA 98052-6399
Phone: 425 882-8080
Web: www.microsoft.com

Test Platform

OS: Windows 2000 Server
Processor: 1.60GHz Intel Pentium IV processor, 60GB disk
Memory: 512MB

Specifications

Platforms: BizTalk Server–
Window 2000 Server/Advanced Server
BizTalk Tools–
Windows 2000
Professional/Server/Advanced Server, Windows XP Professional

Microsoft BizTalk Server 2002 provides an out-of-box solution for integrating internal systems and external business partners. BizTalk Server 2002 is its second generation, succeeding BizTalk Server 2000. It is built on top of a core Microsoft Windows 2000–centric infrastructure running on Windows 2000 Server platforms. BizTalk Server also utilizes SQL Server 7.0/2000 as the repository for BizTalk messaging management, shared queue, tracking, and the orchestration of persistence databases. These databases are created using the BizTalk Server 2002 Installation program.

Quick Install/Setup

To evaluate BizTalk Server 2002, take a look at the BizTalk Server 2002 system requirements (www.microsoft.com/biztalk/evaluation/sysreqs/default.asp). Essentially, you need a Windows 2000 Server (with the latest service pack, of course), SQL Server 7.0SP2/SQL Server 2000, and IE 5.x. In my test environment, I had everything installed on my 512MB RAM/Pentium 4 notebook, but you probably want to install the BizTalk Server and Tools components separately. BizTalk Tools (editor, mapper, messaging manager, etc.) can be installed on Windows 2000 Professional and/or Windows XP Professional platforms as well. If you set up everything from scratch, there are quite a few components to be installed, so give yourself a good 3–4 hours. However, I found that most of the installation process was pretty straightforward and I didn't encounter any errors during setup.

In addition, if you want to integrate BizTalk Server 2002 with Web services you need to install the .NET Framework and the BizTalk Adapter for Web Services and BizTalk Server Toolkit for Microsoft .NET. The former allows BizTalk to initiate integrations with Web services endpoints, while the latter enables developers to develop Application Integration Components (AICs) using the .NET Framework and Visual Studio .NET. Figure 1 shows the BizTalk Server 2002 architecture.

Key Concepts

To help you understand the internals of the BizTalk Server, some key concepts are outlined below.

- Receive functions are starting points for integration. Currently BizTalk Server 2002 supports Message Queue, File, and HTTP Receive functions. Let's say, for instance, you've implemented a File Receive function to listen for files in a particular directory/file share. Whenever a file is created in that directory, the integra-

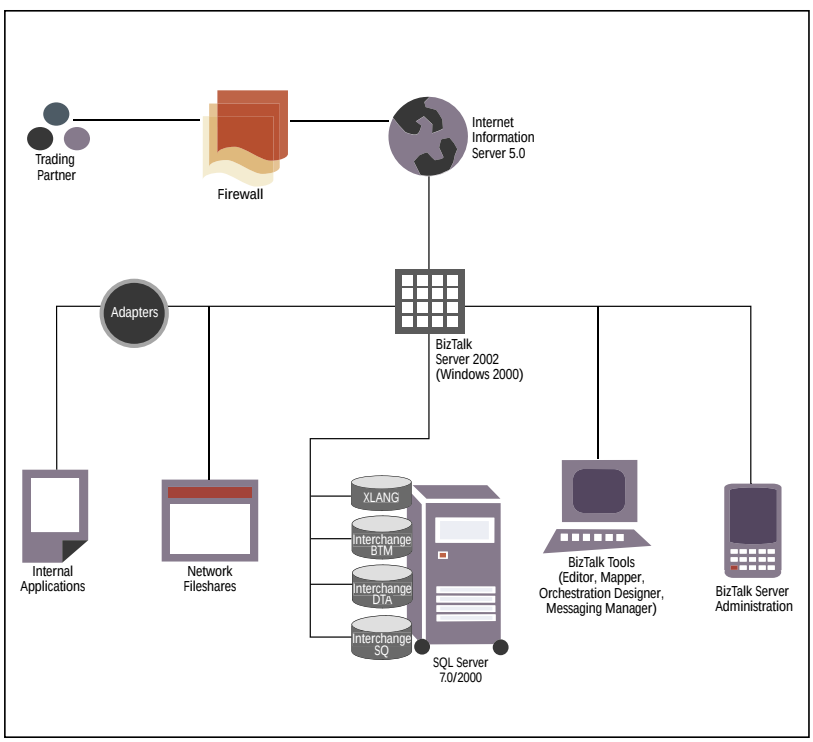


Figure 1 BizTalk Server 2002 architecture

tion business process will be initiated – the process is similar for message queue/HTTP interfaces. In addition, using the IInterchange COM object, COM-aware applications can use an API to initiate integrations.

- Channels provide the transformation, security, and messaging capability to incoming documents. (For example, once you've received the document from your trading partners, chances are that you would like to transform it into your own internal vocabulary.) Channels also enable management of data encryption, digital signatures, and logging features.
- Messaging ports represent document destinations, which can be either applications or XLANG-based schedules. By definition multiple channels can send documents to a single port.
- XLANG Schedule defines the business process. XLANG definitions are developed using the BizTalk Orchestration Designer (a tool developed on top of Visio), which provides a graphical environment to draw an abstract business process using typical flow chart constructs (begin/end, decisions, loops, while, action, fork, bind, abort). It can be implemented by developers to integrate with COM/Script components, message queuing, and BizTalk Messaging.
- Application Integration Components (AIC) are COM objects that receive data for applications.

Component	Key Functions/Highlights
Microsoft SQL Server 7/2000	• Not included in BizTalk but is required for implementation
BizTalk Server 2002	• Core engine • Runs on top of Windows 2000 Server/Adv. Server • Uses W2K Message Queuing infrastructure
BizTalk Editor	• Define/edit documents • Supports XML/EDI/flat files • Supports schema export • Prebuilt schemas for XML/X12/EDIFACT documents
BizTalk Mapper	• Map source to destination formats (XML/EDI/flat files) • Generates XSLT • Supports "Functoids" for complex manipulation operations • Supports custom "Functoids" • Provides testing capability
BizTalk Orchestration Designer	• Microsoft Visio–based visual tool to represent business processes • Built on top of XLANG (an XML markup for business-process interaction design, precursor to BPEL4WS) • Splits tasks into visual design phase and coding phase • Implementation options: Integration with COM components, Windows script component, Message Queuing, and BizTalk Messaging • Provides exception-handling capabilities
BizTalk Messaging Manager	• Trading partner manager • Configure BizTalk Messaging Services – channels, ports, document definitions, document envelopes, and organizations
BizTalk Document Tracking	• Track/analyze documents
BizTalk Server Administration	• MMC Snap-in • Manage BizTalk adapters, queues, and receive functions • Manage document queues • Event viewer
BizTalk SEED Wizard	• Partner enablement • Package and distribute configurations to trading partners
BizTalk Server 2002	• Provides a simpler way (based on hub-and-spoke concepts)
Publish/Subscribe Toolkit	• Use BizTalk objects – ports, channels, receive functions, documents, and organization
BizTalk Server 2002	• Provides capability to develop Application Integration
Toolkit for .NET	• Components (AICs) in .NET
Adapters	• Adapters from Microsoft - MQSeries, SAP, SQL Server & Web services • Other adapters available from third-party providers
Accelerators	• For HIPAA, suppliers, RosettaNet, and financial services • Provide enhancements, tools, and samples to "accelerate" development around specific application scenarios

Table 1 • E-business dialogues

Editions

Microsoft BizTalk Server comes in different editions to support different requirements and scale: the Developer Edition is focused on development purposes only, and the Partner Edition is targeted for quick trading partner enablement and supports integration of internal applications with up to two trading partners. Next is the standard edition, which is designed for small/medium deployments and supports integrating up to 5 internal applications with 10 trading partners. Standard and Partner editions have no support for multiprocessors or clustered deployments. However, the Enterprise Edition, which supports unlimited internal applications/trading partners, fully supports multiple processor deployments and clustering capability.

• • •

We will continue our exploration of BizTalk Server 2002 next month, when we will take a much more hands-on look, actually developing an end-to-end integration scenario using BizTalk Server 2002 and its components.

Resources

- *Microsoft BizTalk Server:* www.microsoft.com/biztalk
- *BizTalk Adapters:* www.microsoft.com/biztalk/evaluation/adapters/default.asp
- *BizTalk Accelerators:* www.microsoft.com/biztalk/evaluation/accelerators/default.asp
- *Microsoft BizTalk Server 2002 Trial Software:* www.microsoft.com/biztalk/evaluation/trial/default.asp
- *MSDN: BizTalk Server:* http://msdn.microsoft.com/biztalk

HITESH@SYS-CON.COM

XML NEWS

Tamino XML Server Provides Extended Support

(Reston, VA) – Software AG, Inc., has announced Tamino XML Server 4.1, the newest version of its award-winning, high-performance data management platform. Tamino is designed for managing XML documents in their native format and is a strategic building block for enterprise information management solutions. Along with a breadth of other new features and enhancements, version 4.1 adds support for the newest XML standards so it works well with a wide variety of IT infrastructures.

Tamino 4.1 offers better performance, significantly shorter implementation times, and investment security for system landscapes. In addition, version 4.1 gives developers a number of new interfaces and upgraded tools.
www.softwareag.com

IBM Announces Multimodal Beta

IBM's Multimodal beta, now available for download at alphaWorks, enables developers to write multimodal applications. The beta includes a Multimodal Toolkit that is based on XHTML+Voice (X+V), a specification comprising XHTML and VoiceXML. X+V represents a unified standard for multimodal interfaces so that applications can be written once and used in different environments – including Web pages, telephones, and handheld devices. This technology is an enhancement to the Opera 7.0 browser, which allows the browser to render Web pages written in the XHTML+Voice markup language.
www.ibm.com/alphaWorks

XAware Announces Eight New Industry Partnerships

(Colorado Springs, CO) – XAware, Inc., has announced partnerships with eight industry-leading application server, data access, XML editor, and e-commerce channel vendors. These partnerships broaden the IT environments, increase the variety of data resources available to XAware's customers, and expand the market reach of XAware's XML-based data integration software. XAware's application server partners are BEA, IBM, Sun, and JBOSS.
www.xaware.com

Stellent Enhances Stellent Content Management

(Eden Prairie, MN) – Stellent, Inc., has announced expanded XML conversion capabilities for Stellent Content Management, enhancing the Stellent system by enabling business users to automatically transform desktop application files to any XML format for application integration.

Using Stellent, business users contribute content in its native file format for conversion to XML, and an XSL is automatically applied based on metadata criteria. Once the new XML document is created, it is automatically validated and made available for integration with other applications.
www.stellent.com

W3C Issues VoiceXML 2.0 as a W3C Candidate Recommendation

Giving voice to the Web, the World Wide Web Consortium (W3C) has published VoiceXML 2.0 as a W3C Candidate Recommendation. Advancement of a W3C Technical Report to Candidate Recommendation is an explicit, public call for implementation. The goal of VoiceXML 2.0 is to bring the advantages of Web-based development and content delivery to interactive voice response applications.
www.w3.org

The 'Oscars of the Software Industry': *WSJ - XML-J* Readers' Choice Awards

(Montvale, NJ) – What Web services and XML products make the difference? Vote for your favorites for the ***Web Services Journal - XML-Journal*** Readers' Choice Awards, often referred to as "The Oscars of the Software Industry." Voting for the 2003 *WSJ - XML-J* Readers' Choice Awards opens March 1, and continues until August 30, 2003. Then come to applaud the winners at Web Services Edge 2003 West, in Santa Clara, CA, October 19-22.

Categories this year include: Best Web Service Security Solution, Best Integrated Services Environment, Best Web Services or XML Site, Best XML Parser, Best XSLT Processor, Best XML Database, Most Innovative Application of XML, Most Overlooked Application of XML, Best Web Services IDE, and Best Web Services Integration Tool. More information is available at our Web site; cast your vote today!
www.sys-con.com

Software Products Certified eBusinessReady

(Lawrenceville, NJ) – eBusinessReady, an industry-neutral software compliance and interoperability testing program under joint partnership of the Uniform Code Council, Inc. (UCC), and Drummond Group, Inc. (DGI), has announced that software products from 11 leading solution providers successfully completed the program's ebXML Messaging 2.0 interoperability testing.

Companies demonstrating interoperability included bTrade, inc.; Centers for Disease Control and Prevention (CDC); Cyclone Commerce; Fujitsu; IPNet Solutions, Inc.; Sonic Software; Sterling Commerce, Inc.; Sun Microsystems; Sybase, Inc.; TIBCO Software, Inc.; and XML Global Technologies.
www.drummondgroup.com

Altova Announces XMLSPY 5 Release 3

(Beverly, MA) – Altova, Inc., producer of XMLSPY, has announced XMLSPY 5 Release 3. XML technologies today are being applied to solve a wide spectrum of enterprise computing challenges, including electronic commerce, document publishing, database integration, and Web services applications.

To maximize software developer productivity in implementing any XML-enabled solution, Altova has added enhanced developer support for various widely used enterprise technologies, thereby accelerating and simplifying XML development. The new version of XMLSPY is available immediately for a free trial download.
www.altova.com

BEA Systems Announces XMLBeans

(San Jose, CA) – XMLBeans makes it easy for developers to access and manipulate XML data and documents in Java. For the first time, developers can gain a familiar and convenient Java object–based view of their XML data without losing access to the richness of the original, native XML structure.

XMLBeans is based on an efficient XML token stream that provides easy navigation of XML data using cursors or XQueries. On top of this, Java class "views" of this data are automatically generated based on an XML Schema description. These Java classes enable easy read/write access to XML information that enforces XML Schema constraints. Because these Java views are based on the underlying XML representation, no information in the original XML is ever lost.
www.bea.com

Altova

http://xmlj.altova.com/wsdl