The face of distributed computing is changing. The client server networks of the 1980s and early 1990s are giving way to a new kind of network built around the Web using open protocols and a new data representation language called XML. This shift is that organizations, instead of focusing on networks and transport, can now shift their attention to data and its role in cementing relationships with customers using the global Web already in place.

This shift in focus from transport to data has been underway since the early 1990s when Tim Berners-Lee augmented an in-place Internet network with a file request protocol called HTTP, a file format known as HTML, and software called a browser for retrieving and displaying HTML. The result was the Web. But it wasn't until the emergence of XML in the mid-1990s, that the Web began to be seen as a vehicle for distributed enterprise computing.

Interestingly, Microsoft was early seeing the writing on the wall and in a deft move, re-engineered itself from a desktop applications and operating system company to one focused on delivering XML-based Web solutions and services. Microsoft's .NET initiative, centered on the notion of software as a Web service, is changing how companies now build enterprise software. Not to be outdone by Microsoft, Sun and its Java juggernaut have fully embraced XML and Web Services as crucial to the success of the enterprise Java initiative.

What's amazing in all of this is that in just a short period of time, XML, a very simple data description language, in combination with the same simple protocols we use to surf the Web, has changed the name of the game in distributed computing.

What is XML

XML, the Extensible Markup Language, is a metalanguage (literally languages) defined by the World Wide Web Consortium (W3C), one driving the push to open Web standards.

The story behind XML is very much the story of the Web. Just as the Web has influenced almost every aspect of our lives, from work, to
so has XML, a universal data description language, affected the distributed computing landscape. The result has been the opening up of new opportunities for advantage by moving XML using accepted Internet standards.

In its simplest sense, XML is a set of rules and guidelines for describing structured data in plain text rather than proprietary binary representations. However, as goes beyond its technical specification. Since its standardization by the W3C, XML has been the driving force behind numerous other standards and vocabularies that are forging a fundamental change in the software world.

Much of XML's success stems from its simplicity. Technically, it's a language based on the insertion of tags to help describe data. It's more than just tags. To see what we mean by this, let's look at a simple XML document that describes the purchase of 100 shares of Zorkon stock.

```xml
<?xml
<order transaction="buy">
  <company>Zorkon Ltd</company>
  <shares>100</shares>
</order>
```

There are several important basic points to observe about this XML:

-- It's readable. Except for the first line which is required for all XML documents, it's easy to figure out what's going on. It doesn't take a degree in computer science to figure out that the data relates to the purchase of 100 shares of stock.

-- There are three elements: order, company and shares.

-- The order element contains the attribute transaction.

-- Data may be stored in either elements or attributes.

-- Elements and attributes are named to give the data meaning.

-- Start tags (e.g.) and end tags (e.g.) define elements that are the basis for tree-structured representations of documents.

-- Elements can contain text data and/or other elements.

Although there is more to XML than the few simple rules listed above, its essence can be grasped and understood quite easily. As a simple data description language, XML forms the core of a family of supporting technologies that can be combined in a variety of ways to meet a wide range of requirements. Figure 1 illustrates XML and its technology family.
Much of the XML advantage is that widespread industry support exists for XML. Numerous tools and utilities are available with Web browsers, databases, and operating systems, making it easier and less expensive for small and medium-sized organizations to import and export data in XML format. A large family of XML support technologies is available for the interpretation and transformation of XML data for Web page display and report generation, and major relational database vendors now have the native capability to read and generate XML data.

This widespread support for XML has opened the door to creative ideas about delivering XML over standard web protocols such as HTTP, FTP, or Simple Mail Transfer Protocol (SMTP).

**XML: The Three Revolutions**

To understand XML's impact, it's helpful to look at XML in the context of three revolutions centered on XML and the Web. These include a revolution in how we think about and use data, a revolution in software architecture that has moved us from tightly coupled networks to a more loosely coupled, open Web space, and a revolution in how we build complex software.

**The Data Revolution**

Prior to XML, data was very much proprietary, closely associated with applications that understood how data was formatted and how to process it. Now, XML data vocabularies provide alternatives to specialized Electronic Data Interchange (EDI) solutions by facilitating B2B data exchange and playing a key role as infrastructure for distributed computing. XML's strength is its data independence, not tied to any programming language, operating system, or transport protocol. In the grand scheme of distributed computing, this is a radical idea. It doesn't require lock-in to programmatic infrastructures to make data available.
platforms. In effect, as Figure 2 illustrates, data is free to move about globally without the constraints imposed by tightly coupled transport dependent architectures. XML's sole focus on data means that a variety of transport technologies may be used to move XML across the Web. As a result, protocols such as HTTP have had a tremendous impact on XML and have opened the door to alternatives to CORBA, RMI and DCOM, which don't work over TCP/IP. XML does this by focusing on data and leaving other issues to supporting technologies.

**The Data Revolution**

The architectural revolution surrounding XML is reflected in a move from tightly coupled systems based on established infrastructures such as CORBA, RMI and DCOM, each with their own transport protocol, to loosely coupled systems riding atop standard Web protocols such as TCP/IP. Although the transport protocols underlying CORBA, RMI, and DCOM provide for efficient communication between nodes, their drawback is their inability to communicate with other tightly coupled systems or directly with the Web. Loosely coupled Web-based systems, on the other hand, provide what has long been considered the Holy Grail of computing: universal connectivity. Using TCP/IP as the transport, systems can establish connections with each other using common open-Web protocols. Although it is possible to build software bridges linking tightly coupled systems with each other and the Web, such efforts are not trivial and add another layer of complexity on top of already complex infrastructure. As illustrated in Figure 3, the loose coupling of the Web makes possible new system architectures built around message-based middleware or less structured peer-to-peer interaction.

**Figure 2. The Data Revolution**

**The Architectural Revolution**

The architectural revolution surrounding XML is reflected in a move from tightly coupled systems based on established infrastructures such as CORBA, RMI and their own transport protocol, to loosely coupled systems riding atop standard Web protocols such as TCP/IP. Although the transport protocols underlying CORBA, RMI and DCOM provide for efficient communication between nodes, their drawback is their inability to communicate with other tightly coupled systems or directly with the Web. Loosely coupled Web-based systems, on the other hand, provide what has long been considered the Holy Grail of computing: universal connectivity. Using TCP/IP as the transport, systems can establish connections with each other using common open-Web protocols. Although it is possible to build software bridges linking tightly coupled systems with each other and the Web, such efforts are not trivial and add another layer of complexity on top of already complex infrastructure. As illustrated in Figure 3, the loose coupling of the Web makes possible new system architectures built around message-based middleware or less structured peer-to-peer interaction.
XML plays a key role in this new architecture for distributed computing through a new XML protocol language called SOAP, the Simple Object Access Protocol. Although SOAP has the word object as part of its acronym, the protocol is not object-oriented in the sense of C++ or Java. Rather, SOAP simply defines a set of XML tags for moving XML data around the Web using standard Web protocols, accomplishing in one simple initiative what client-server computing had been trying to do for over a decade. Associated with this ability to move data freely across the Web is the rise in the use of messaging servers and software that sit between conversational participants. These servers, supporting what is known as Message Oriented Middleware, are playing an increasingly important role in the new extended enterprise by providing guarantees of delivery and the ability to broadcast communications to multiple recipients.

A case in point is the Web. After decades of attempts to build complex infrastructures for exchanging information across distributed networks, the Web emerged from an assemblage of foundational technologies such as HTTP, HTML, browsers and a longstanding networking technology known as TCP/IP that had been put in place in the 1970s.
One aspect of the new extended enterprise is the emergence of Web services, an ambitious initiative that is moving the Web to new levels of B2B (software-to-software) interaction while trying to fulfill object technology's promise of reusable components from a service interface. For some, Web services represent the next evolutionary step for the Web, extending it from a network that provides services to humans to one that provides services to software looking to connect with other software.

A Web service can be anything from a movie review service, to a real-time weather advisory, to an entire hotel- and airline-booking package. The Web services technical infrastructure ensures that services even from different vendors will interoperate to create a complete business process. Web services take the object-oriented vision of assembling building blocks to the next level. With Web services, however, the emphasis is on the assembly of services that may or may not be built on object technology.

From an enterprise perspective, a Web service is both a process and a means of connecting to software exposed as services over the Web. By using protocols such as HTTP and SOAP, developers can concentrate on what data to exchange instead of worrying about how to get it from point A to point B. To this end, SOAP also defines an XML envelope to carry XML data, as well as a convention for doing remote procedure calls. This RPC capability allows Web service providers to advertise "call me here" so that a program (not a human) will be able to do so without concern for language or platform.
As illustrated in Figure 5, XML, SOAP and Web Services define a new landscape for distributed computing that includes XML as the data, SOAP and HTTP as the protocols for moving data across the Web, and Web Services protocols such as UDDI and WSDL for the discovery and connection to those services. Playing a central role in this automatic Web-based discovery and lookup is a shift to a central repository that maintains yellow pages for category search, white pages for direct name search, and green pages for details about making a connection. Whether these repositories exist behind or outside firewalls is up to the application implementation. However, the basic idea that companies or departments with services to offer can register and be discovered by clients seeking those services, is changing the face of the enterprise. While it remains to be seen how all this plays out, the bottom line is that we’re moving into a new era of distributed interaction, driven by XML.

Summary

The simplicity of XML in combination with the Web has opened up new possibilities for moving data and for building new application architectures centered around common Internet protocols. Some of the changes brought about by XML include:

-- A reduced dependence on proprietary data formats for applications
-- A new way to do B2B data exchange using XML instead of the formats defined by traditional EDI systems
-- A shift from relying on tightly coupled systems such as CORBA, RMI, and DCOM to a more loosely coupled Internet-based framework centered around XML
-- A change in focus from object-oriented to service-oriented software

-- The emergence of Web services as technology for discovering and Internet-based services

-- A move away from monolithic applications that attempt to do it all software model that derives new capabilities from the combination of scope, software components.

Forum

Frank Coyle is a senior lecturer and director of the Software Engineering Program at Southern Methodist University in Dallas, Texas. He received his BS from Fordham College, an MS from Georgia Tech and a PhD from SMU. His recent book, *XML, Web Services and the Data Revolution* is published by Addison-Wesley.

Ubiquity welcomes the submissions of articles from everyone interested in the future of information technology. Everything published in Ubiquity is copyrighted ©2001 by the ACM and the individual authors.