XML: SOLVING BUSINESS PROBLEMS

WHITE PAPER

by George Reese

CUTTER CONSORTIUM, 37 BROADWAY, SUITE 1, ARLINGTON, MA 02474-5552, USA
Tel: +1 781 641 5110, Fax: +1 781 648 8707
consortium@cutter.com; www.cutter.com/consortium/

CUTTER CONSORTIUM

ABOUT THE CUTTER CONSORTIUM

The Cutter Consortium was founded to help organizations forge solutions to the IT challenges they face. Its mission is to serve the user community by helping enterprises create, implement, and maintain IT systems and IT organizations that support their business objectives.

As a completely independent entity that does not accept any special funding from vendors, the Consortium is beholden to no one and dedicated to providing completely objective, unbiased information to its clients. An important feature of the Consortium is its faculty and teams of Senior Consultants — a distinguished group of internationally recognized experts committed to providing top-level, critical, objective advice.

The Cutter Consortium is affiliated with Cutter Information Corp., a leading information provider since 1986. Cutter has 20,000 clients located throughout the world. These professionals rely on Cutter newsletters, reports, research studies, databases, and electronic services to obtain the advice, news, and analysis their jobs demand. As a smaller information provider, the Consortium will customize its services to meet each client's needs.

THE CONSORTIUM'S DISTRIBUTED COMPUTING ARCHITECTURE/ E-BUSINESS ADVISORY SERVICE

Cutter Consortium's Distributed Computing Architecture/E-Business Advisory Service is the only service that puts the leading experts to work on the architecture challenges facing your organization. The service not only focuses on middleware and associated operating system considerations, but on components and application frameworks. It also evaluates the middleware products and component tools that are so critical to making distributed systems work and to developing reuse-based organizations.

Through our Advisory Service offerings, and our custom consulting and on-site assessments, you gain direct access to a team of senior consultants, including Doug Barry, Ted Burghart, David Frankel, Curt Hall, Michael Guttman, Brad Kain, Jean Pierre LeJacq, Jason Matthews, James Odell, John Rymer, Kent Seinfeld, Roger Sessions, and Ed Yourdon, giving you the confidence that comes from relying on the best minds in the industry.

Of equal importance, Cutter Consortium offers clients maximum flexibility by offering a wide range of products and services that can be customized to meet your organization's specific needs. The Consortium forms a close partnership with its clients. We offer a variety of exclusive resources including access to some of the best thinkers in the business. Information is customized and provided in numerous formats designed to meet your information needs.

The stakes are high with a move toward distributed computing architectures and e-business. Every option you select will be expensive. Cutter Consortium's comprehensive approach to distributed computing architecture and its emphasis on objective assessment of architectural options and on cost-effective, flexible solutions will help you make the best decisions given your organization's specific strengths and needs.

FOR MORE INFORMATION

To learn more about the Cutter Consortium's Distributed Computing Architecture/E-Business Advisory Service, call +1 781 641 5118, send e-mail to consortium@cutter.com, or visit the Cutter Consortium Web site: www.cutter.com/consortium/.

Distributed Computing Architecture/e-Business Advisory Service Executive Report is published by the Cutter Consortium, 37 Broadway, Suite 1, Arlington, MA 02474-5552. Tel: +1 781 641 5118 or, within North America, +1 800 964 5118, Fax: +1 781 648 1950 or, within North America, +1 800 888 1816, E-mail: consortium@cutter.com, Web site: www.cutter.com/consortium/.

XML: Solving Business Problems

White Paper

by George Reese

Important technologies inspire innovation, motivating technologists and entrepreneurs to both create new roles for the technology and use it to solve old problems. The Internet is the most glamorous example of such a technology. Among the innovations inspired by the Internet are network-centric technologies such as Java and XML; however, these innovations are wrapped in a veil of hype. The benefit of the hype is that it draws people in, bringing about creative work with the technology. The downside is that it leads to misguided attempts at applying the technology to the wrong problems. For an observer, sorting out the valid applications of a hyped technology from the misapplications can be hard work.

Java is a good example of how an exciting new technology can be misapplied through hype. Java's origins predate the Internet in the form of a set-top box programming language called Oak. This form did not go very far until Sun Microsystems repackaged it as a cross-platform programming language for making Web pages interactive. However, because of incompatibilities in Web browsers, this application of Java never quite worked. By 1996, Java was positioned as an enterprise programming language, a niche in which it has enjoyed tremendous success. At the same time, Java is again being positioned as an embedded-systems language. In fact, throughout its history Java has been presented as the solution to just about every kind of programming problem. Although it has proven itself in many of these environments, it has also proven itself nonfunctional (and even nonsensical) in others.

The boundless potential evident in XML has created a similar level of hype that leaves decisionmakers wondering where XML fits in the enterprise. Early Java adopters who took a wrong path with the language often dismissed Java as the problem rather than addressing their own misuse of the technology. The potential to make the same mistake with XML is ever present. The challenge for IT decisionmakers is to familiarize themselves with XML's essential characteristics and be able to identify problems for which XML is the best answer.

This White Paper looks at XML as a business technology, identifying the core strengths of this empowering technology.

By understanding these core strengths, IT decisionmakers can better weed out the more bleeding-edge or questionable applications of XML and focus on the business problems for which XML is ideally suited.

THE ESSENCE OF XML

According to Boris Feldman's XML FAQ (www.xml-zone.com/xmlfaq.asp), the Extensible Markup Language (XML) is a "data format for structured document interchange on the Web." Its purpose is to support the sharing of content among diverse entities that may impose vastly different interpretations upon that content. More simply, it is a tool for representing the essential *structure* of content without dependencies on more superfluous issues such as presentation, storage, or application processes.

The popular tools for representing content — such as HTML, Microsoft Word, and relational tables — are all more or less biased to some particular interpretation of that content. HTML files, for example, support the cross-platform exchange of content for display over the Internet. An HTML file displaying a product for sale on an e-commerce site embeds directives for the display of the product along with the product data. The structure of that HTML document is no different for the display of this product than it is for any other set of data to be displayed in the same manner. As a result, the content of an HTML file is not useable in a context outside of display for the Web.

XML as a Meta Language

Unlike HTML, XML is not a "true" markup language. Instead, it more closely resembles the Standardized Generalized Markup Language (SGML) in that it is a meta language for describing other markup languages. XML is, in fact, a simplified dialect of SGML. Although SGML has all the features it needs to do what XML does, it is generally considered too complex for widespread adoption as a standard for representing structured content.

The World Wide Web Consortium (W3C) is the independent standards organization responsible for a variety of Internet standards, including HTML and XML. The current specification is available on its Web site at www.w3.org/xml. Because XML is a meta language, just "knowing" XML is relatively useless. XML's power is derived from the specific markup languages it is used to define. HTML 4.0 is an example of one such language.

Although HTML predates XML, the HTML 4.0 specification redefined HTML as a subset of XML. This redefinition is possible because HTML, like XML, was a subset of SGML. The task of HTML is simply to represent content — independent of its underlying nature — for display on the Web. As a result, a Web browser does not need to be concerned with what sort of data it is representing. A Web browser will display the text between the `<EM>` and `</EM>` (emphasis) tags exactly the same way, regardless of whether that data is the price of a toy, the warning for a drug's possible interactions with other drugs, or the score for the home team in a basketball game. This agnosticism about the meaning of the underlying content is essential for Web browsers.

Beyond HTML

Although HTML is an excellent standard for representing content for display on the Web, it makes a poor standard for the canonical representation of the data behind business concepts — in other words, an HTML document does not serve as the authoritative source for the information it contains. No one really expects other applications to be able to read `<EM>$12.95</EM>` and understand it as the price for a product. HTML, therefore, cannot be used as a mechanism for interchange of product data, only for the interchange of data for display.

Beyond the issue of separation of content from display is the fact that HTML is good for the display

of content *only on the Web*. HTML is hopelessly (and necessarily) too incomplete to support the more complex display features of tools such as Microsoft Word, Adobe FrameMaker, or Quark XPress. In contrast, XML not only supports the ability to define markup languages designed to represent the structure of content, but it also supports the ability to translate between two XML languages. A product represented in one form of XML can be translated to another form of XML, such as HTML, to enable the product to be shown on the Web.

Canonical Representations

I mentioned that HTML was a poor standard for the canonical representation of content. In most organizations, the canonical representation of business data is in the form of rows in relational database tables. Among the more questionable applications of XML is an attempt to replace the relational database table as the underlying canonical representation of all data. The relational database not only serves purposes that XML doesn't, but most businesses also have a heavy investment in their relational infrastructure. However, XML makes for an excellent application-independent canonical representation of data.

Relational databases tailor data for a specific use (more on this idea later). As a result, the data model that supports one business system is often a very poor data model to support some other system. XML enables the legacy systems responsible for maintaining some data set to continue to do so while providing a canonical XML view for external systems. By standardizing on an XML dialect for the canonical representation of data, a business makes all of its systems independent of the actual data model of the system of record for any particular data set.

Figure 1 shows XML for a hypothetical set of product data represented in a mythical dialect of XML, which is designed to represent product information. In this example, you can see that the price is clearly identified by the `<price></price>` tags. As a result, any application that cares about the price of a product can grab that information from the XML representation of that product.

Figure 1 is actually still a poor candidate as an XML dialect for representing product information canonically because it lacks the force of standardization. Only applications written specifically to understand this mythical XML dialect can make use of it without an Extensible Stylesheet Language Transformation (XSLT). To share this product information with other applications, the product information must either be represented in an industry standard for capturing product information or have an XSLT transformation written for it. XSLT transformations are covered in the next section; however, it is less work to simply use an industry standard.

Transformations

To avoid maintaining both a canonical source as well as multiple Web pages with the same data, W3C maintains a standard for transforming XML documents of one type into documents of

```
<?xml version="1.0"?>
<product>
   <name>Mr. Froggy and Friends</name>
   <description>
      This is the famous book that tells of the wacky
      adventures of Mr. Froggy
   </description>
   <property>
      <propName>ISBN</propName>
      <propValue type="text">XXXXX</propValue>
   </property>
   <price>$5.95</price>
</product>
```

Figure 1 — An XML document containing product information.

another XML type. This standard, XSLT, is an XML dialect based on XML's stylesheet language: XSL.

The most common kind of translation is from a content-oriented XML dialect into a display-oriented XML dialect like HTML. Using XSLT, business data stored in XML can easily be displayed on the Web without having to maintain the data both in XML documents and in HTML documents. The XSLT standards are available on the Web at www.w3.org/TR/xslt.

Linking

One of the more powerful features of HTML is the ability to link documents. Even so, HTML linking is fairly weak. Specifically, HTML supports only one kind of linking: a link via the <A> tag that instructs a browser to traverse from the linking document to the linked document. In a Web context, this might seem like all you could want, but publishing and other document domains demand much more.

XML has a linking standard that comes in two pieces: XLink and XPointer. The XLink standard specifies how an author inserts a link in an XML document. XML requires a standard like XLink because, unlike HTML, any number of tags can be "linker" tags. XLink simply defines a syntax that XML authors use for indicating a link. More importantly, XLink enables an author to support many different kinds of links, including footnotes, annotations, and hypertext links.

XPointer governs how your links point to other documents or document fragments. It supports the conventional syntax common to HTML; it also supports the ability to point to document elements unthinkable in HTML. In HTML, if you want to link to a part of someone else's document, that person would have to know ahead of time that you might want to link to that part and that individual would have to provide you with a named marker in the HTML document. Consider this report in HTML on the Web. If a Web author wanted to create a link to this section of the report from the author's Web page, I would need to place a marker in my HTML code at this section's header, and the other author would have to know the name of that marker. XPointer removes this requirement from document authors. An XML document author can link to any point in any other XML document without the need for markers placed by the referenced document's original author.

Some XML Standards

Because XML makes no effort to define what tags make sense, people with common requirements for structured content generally get together and define something called a document type definition (DTD). A DTD is an XML document that identifies which tags are valid for a specific kind of XML document. For example, a DTD for Figure 1 would specify that an XML document of type <product> can contain <name>, <description>, <property>, and <price> tags. Anything else between the <product> and </product> tags would be considered illegal. The following is a brief list of a few XML dialects with standard DTDs in use today.

- **The First Retail Markup Language** (FRML) defines the structure in which retail information may be kept. The goal of FRML is to enable various parts of the retail chain to be able to share retail information.
- **The Music Markup Language** (MML) provides a structure for music content. The goal of MML is to capture the structure of music so that it can be presented differently for different environments. One application might interpret an MML document by displaying it visually; another application might interpret that same MML document by playing the notes.
- **The Open Financial Exchange** (OFX) is an XML schema for financial content. It is most commonly used to transmit information about financial transactions between financial institutions and financial software such as Microsoft Money and Intuit Quicken. When Quicken "talks" to a bank's Web site to get your latest bank

statement, it transmits that data in OFX. As a result, the bank does not need to worry about what kind of software the client is using, and the client software does not need to know anything special about the bank to which it is connecting.

- **XML/EDI** is an attempt to encapsulate Electronic Data Interchange (EDI) using XML. EDI is a common standard for the sharing of data among businesses. The purpose of XML/EDI is largely to create a migration path for EDI-based business systems to an XML-oriented world.

BUSINESS ISSUES

With a basic understanding of what XML has to offer, it is time to turn to the business problems to which it can be applied, including relational database limitations, difficulties in managing digital assets, and extranet integration for e-business.

This report discusses how companies have a significant investment in relational technology for business data storage. Although relational technology is extremely important to business computing, it brings with it some real problems. In addition, the migration from client-server to distributed Internet architectures in the past decade has created a vastly increased demand on business systems to be able to interoperate with one another.

Distributed Computing in the Enterprise

The past decade has seen business systems begin to evolve from client-server to distributed computing architectures. Nevertheless, most businesses are still made up primarily of client-server and mainframe systems. One of the key challenges facing businesses is facilitating the integration of numerous systems with distinct architectures. Figure 2 illustrates a modern business computing infrastructure.

Under this infrastructure, the relational database is the heart of the enterprise. A business probably has numerous databases, ranging from isolated departmental Microsoft Access databases to client-server systems to enterprise-wide distributed business object-persistence repositories. In this particular infrastructure, I have included a few components found only in the most modern enterprises. The most significant of these components is digital asset management.

Digital asset management involves treating the business's digital assets (images, video, audio, etc.) on par with its physical assets. In general, most companies treat their digital assets as second-class assets. The following are some symptoms common to companies that lack a digital asset management strategy:

- Images key to the corporate branding strategy that were initially made available on a central file server now exist all over the place, and no one is sure which images are the most current.
- Interesting or important PowerPoint presentations remain on the author's laptop. Only those who specifically request the presentation from the author get copies.

The infrastructure in Figure 2 looks toward an organized digital asset management strategy. As we will discuss later, XML is a key component in a mature digital asset management architecture.

The Relational Database

The center of the enterprise is the relational database. A software product generally signifies that it is database-aware by being branded with the "enterprise" moniker. The first step in Sun's enterprise-enabling of Java was the introduction of the JDBC database access API to the language. In short, if you ignore the database when discussing enterprise computing, you are simply not discussing enterprise computing.

Under both mainframe and client-server architectures, the relational database is critical to the application. A change to the data model requires extensive modifications to and regression testing of the rest of the system. Because of this tight coupling of the data and the business processes it supports, two separate systems

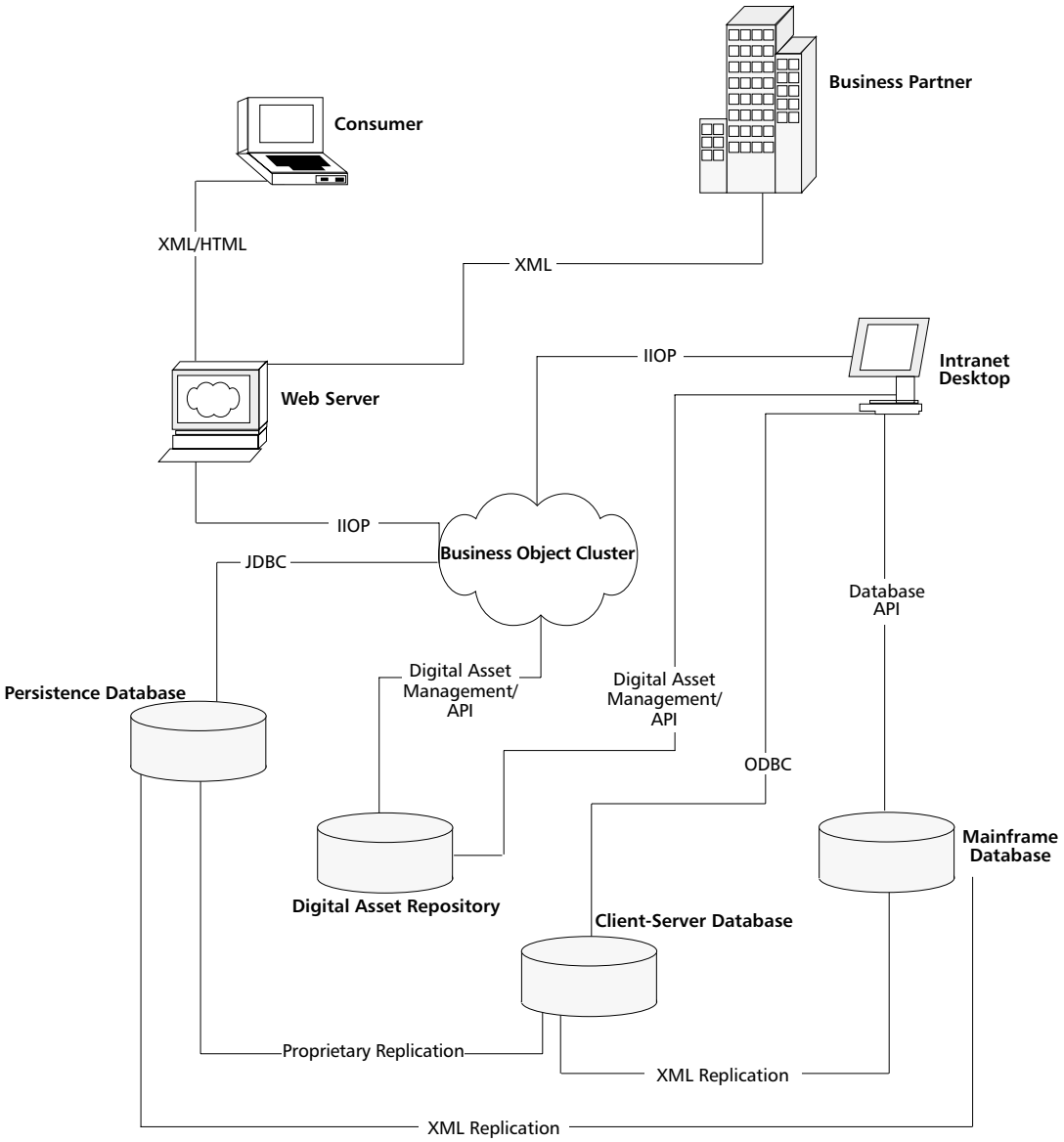

Figure 2 — Example of a modern computing infrastructure.

with similar data can often require two very different data models to support them.

An example of this is a company with separate order-entry and order-fulfillment systems. Both systems store data about the company's products, but each has a different data model for storing that information. Figures 3 and 4 show how these two data models might differ in a hypothetical business.

The problem introduced here is that the disconnect between the two data models demands some level of effort to get order information from the order-entry system into the fulfillment system. If you are using the same database engine for both systems, your vendor's replication technology will likely take care of this problem. Two systems do not always, however, use the same database engine.

If you export the problem to the Web, you will find the problem compounded when trying to display order status to a customer

over the Internet. Which system do you get the data from? How do you turn relational tables into HTML pages? This transformation almost always requires a programmatic solution in the form of CGI, PHP, Active Server Pages, or Java servlets.

Objects and Data

Part of the problem with the integration of relational business systems is that relational systems focus on specific business processes and the data required to support those processes, but object-oriented software engineering moves the focus from the data to the static business concepts the data represents.

Not only do relational systems have problems integrating, they also tend to have trouble evolving as a business evolves. Among the reasons for these problems is the focus on business processes. By their very nature, business processes are very dynamic. They are specific to each business as it exists at a given point, and they change as the business climate changes. Looking at the travel industry, the way one booked a ticket on a stagecoach in 1900 is very different from the way one booked an airline ticket in 1950. That, in turn, is very different from the way people book tickets using the Web today. The travel industry as a general rule is having a difficult time adapting to these changes.

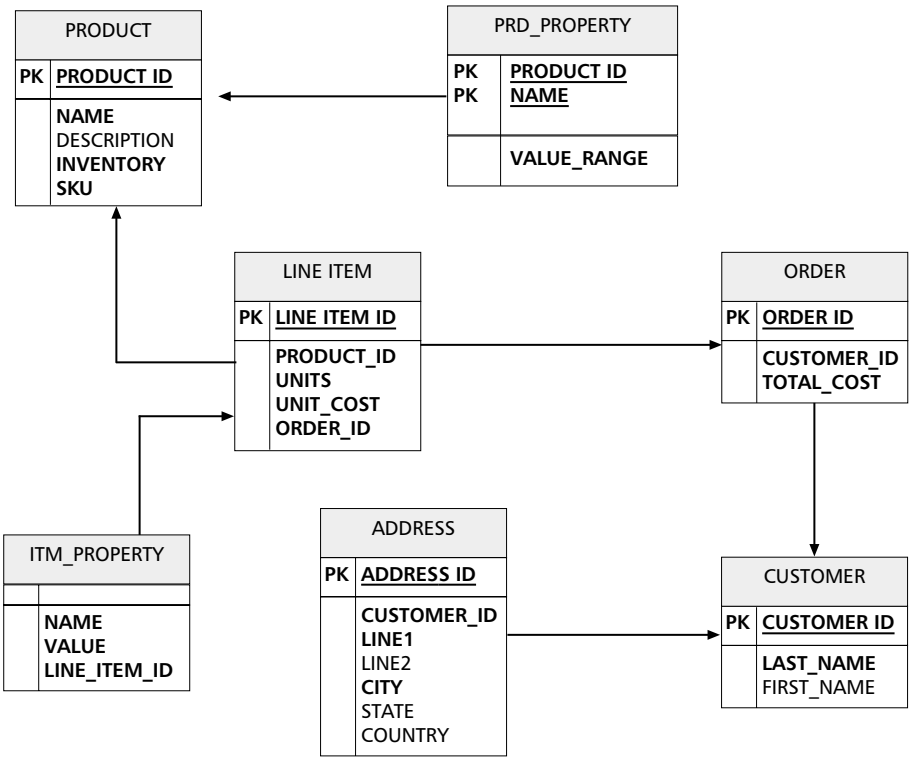

Figure 3 — An order-entry system.

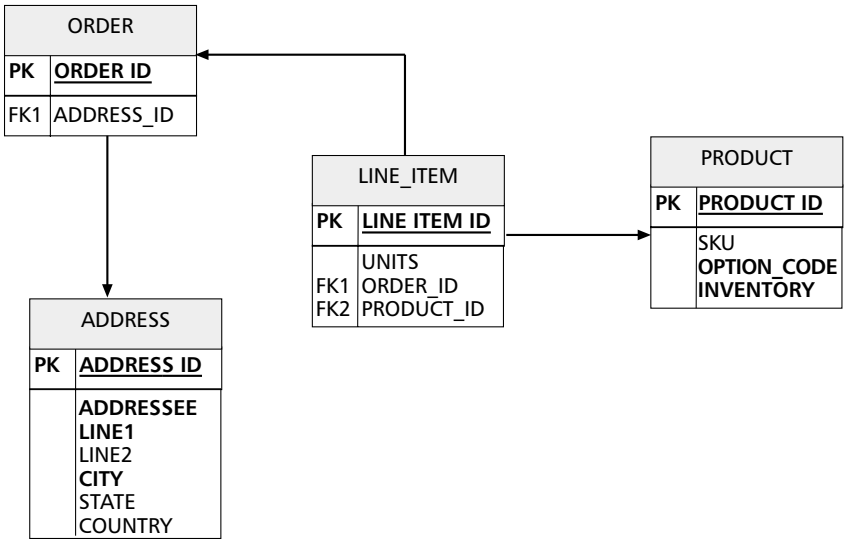

Figure 4 — A fulfillment system.

Over the past century, however, very little has changed in what makes up the travel industry. Although we travel in planes instead of stagecoaches, the essential traits a travel agent cares about — booking seats on some sort of transport — have not changed at all. Only the "how" has changed. Moreover, these business concepts are static across an industry and not specific to any particular business in that industry. A system that is capable of capturing the timeless concepts of a business and gluing them together to suit the whims of the current business climate provides a clear strategic advantage over other systems. Object-oriented software engineering is the philosophy behind this approach to software development.

Most distributed systems are object-oriented to some degree. Technologies such as CORBA wrap business systems in object-like interfaces that represent the static business concepts in those systems. More modern Enterprise JavaBean (EJB) systems capture those concepts in central, shared, object-server clusters to be used and reused by multiple applications.

Two difficult problems exist with object-oriented systems. First is the old problem of integration. Although object-oriented systems can capture business concepts in a simple, shared environment and thus minimize the number of systems requiring integration, two object-oriented systems may use very different distributed computing technologies or component models. Another problem is that capturing business concepts in a centralized, object-oriented system may be overkill for smaller problems. A business may want to capture the data behind a business concept independent of any particular business processes, without building a huge distributed computing solution.

Digital Asset Management

A key part of enterprise computing ignored by most businesses today is digital asset management. Digital asset management is more than managing multimedia data such as audio, video, and images; it is any kind of related computer data — Word documents, HTML files, Netscape profiles, software executables, etc. Digital asset management presents two key problems that fit with XML's strengths.

The first problem relates to the structuring of documents. Most businesses do not generally see this as a huge problem. For some key industries, however, the solution to this problem may determine who succeeds in the Internet era. If you look at a Word file, for example, you must take it as a whole document formatted for Word. You cannot automatically decompose it and analyze it and reuse pieces of the document in new documents. As with HTML, this problem is caused because Word combines style information along with the document content.

If you can manage your documents in a style-free document, as shown in Figure 1 on page 3, those documents can be reorganized and restyled in numerous ways while requiring maintenance in only a single location. Consider the problem of textbooks for college courses. It is not unusual in the humanities for a course to use two or three chapters from a handful of textbooks. Ideally, a professor could ask a publisher for the individual chapters to be put together into a custom textbook. The only way a publisher could easily offer this service, however, is if the publisher had the texts stored in some sort of structured format.

The second problem for digital asset management is the need to integrate different types of assets into a single structured document. Any document format allows you to embed images and such inside a document. It is not easy, however, to find a document format that integrates with a digital asset management strategy, so that an image is not actually embedded in the document, but instead referenced in the digital asset management repository by the document.

E-Business

A common theme throughout all of the problem domains mentioned so far is the problem of integration.

Systems integrate with each other at three different levels: the intranet, the extranet, and the Internet. The focus of this report so far has been on the intranet — those systems over which a business has full control. Intranet integration generally involves the integration of a finite number of well-understood systems whose operational parameters, including the database engine and operating system, are within the control of the integrators.

However, a business has very little control over the standards of its partners. At this level of integration (the extranet level), the challenge is to agree on a standard for data interchange among businesses. Creating those standards becomes unmanageable if the number of businesses involved grows too high. At that point, industry-wide standards are required.

Business-to-consumer integration (Internet integration) compounds the lack of platform control with the problem of too many systems. A bank cannot demand that each of its customers run Windows 98, nor can it reasonably impose the use of Quicken. It is equally impractical — and potentially insecure — for the business to devise a new communication scheme for each customer with a new financial software package.

Until the Internet age, the demand did not really exist for integration much beyond intranet integration. Extranet integration existed, but it did not have the importance it has today, and it often came in the form of expensive solutions such as EDI. In an Internet-oriented business, however, all systems must be capable of integrating at all levels. They must no longer assume a safe network environment. Formerly closed business systems must now be built upon open Internet standards.

SYSTEM INTEGRATION WITH XML

One of the major problems XML addresses is the integration of systems at all levels. XML enables a business to define rules for capturing business concepts that can span an enterprise or an industry. With a standard format for capturing a business concept, a system can rely on a single format for interacting with all other systems. It no longer cares what operating system the other systems use, what database engine they use, or what data model they use.

The central task behind the successful use of XML in the integration of business systems is this: the definition of an XML vocabulary that is capable of capturing the business concepts inherent in an enterprise or industry. If the definition is created by a single business in isolation, that business will not realize the full power of XML in supporting integration with other businesses or even with consumer applications. Successful business-to-business or business-to-consumer integration demands a concerted standardization effort within an industry. In many cases, this may require the cooperation of competitors who may be seeking to use XML to gain a competitive advantage.

Figure 5 illustrates a process flow that involves XML as the integration piece binding a legacy relational system to an external system. In this diagram, a business partner requests information about a specific product via HTTP, invoking a Java servlet. The servlet reads the database and formats the data according to the DTD for the industry. The business partner then parses the resulting XML for storage in its database according to its data model.

Because the XML dialect in question is an industry standard, the external system does not need to know anything about the first business's internal system. This independence of the two systems from one another has two key advantages:

1. The first business needs to write only one export routine to make the data available to all of its business partners. Similarly, the second business needs to write only one import routine to import similar data from all of its business partners.
2. The first business does not need to give out any potentially sensitive information about the internal architecture or design of its system.

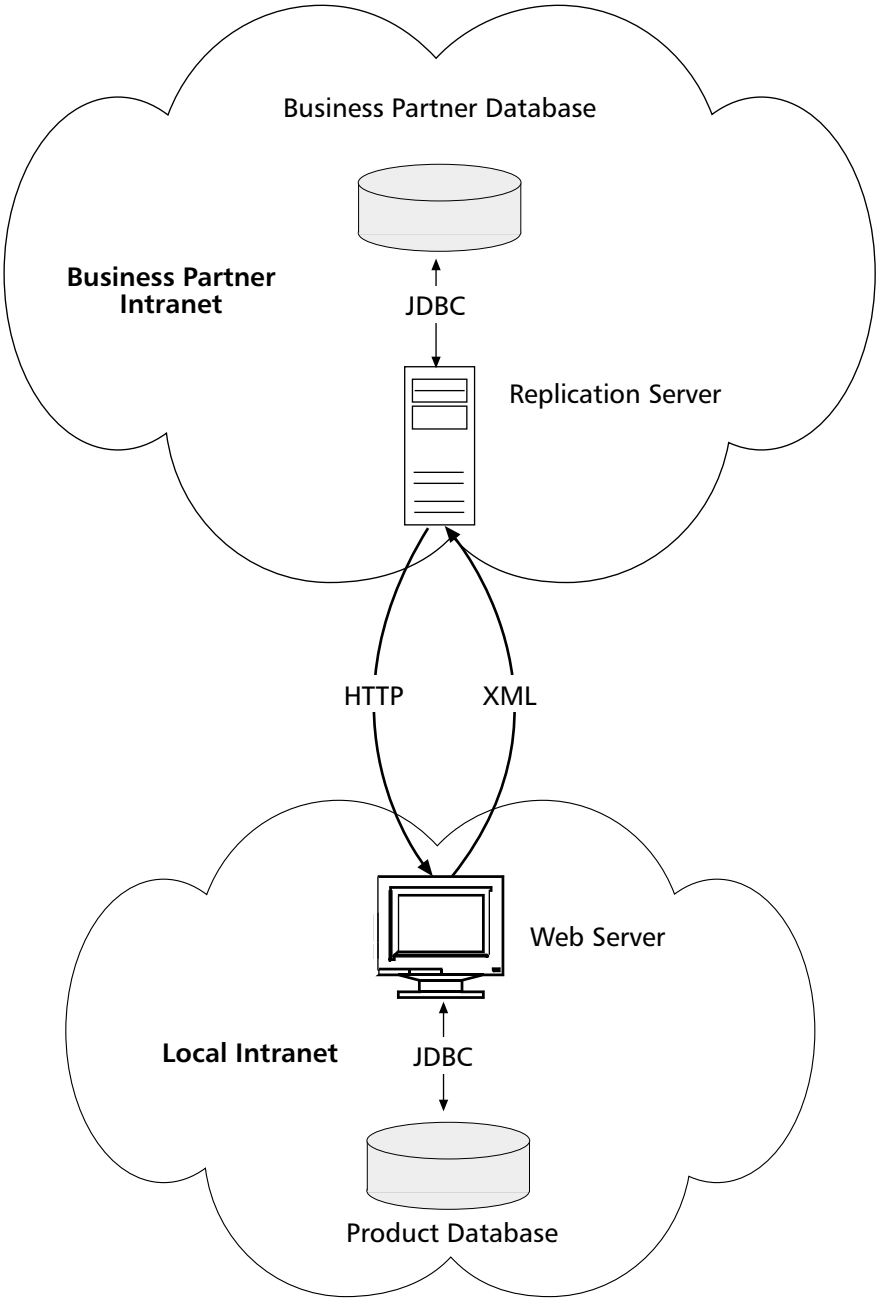

Figure 5 — XML binds a legacy system to an external system.

The net result is that much less work is required to integrate systems and there is greater security that trade secrets are not being exposed to business partners.

DTD Definition

The act of defining a standard XML vocabulary is called DTD definition. As mentioned earlier, the best DTDs are those that have been accepted as a standard by an entire industry. Of course, it is quite difficult to bring together competitors in a fiercely competitive industry to discuss a tool that might potentially provide a strategic advantage. Agreement on a standard can be even more elusive. Thus, it often makes sense for competitive industries to work through an independent, nonprofit standards organization in the definition of DTDs.

The act of DTD definition mirrors the process of good class modeling in object-oriented software engineering. Specifically, the task of DTD definition involves determining what business concepts populate a given industry and identifying the essential characteristics for those business concepts. The Unified Modeling Language (UML) is an excellent tool for expressing the design of those concepts in the DTD definition process. It is a modeling language that helps people visualize classes in an object-oriented framework. Figure 6 is a class diagram for a hypothetical banking industry XML standard.

DTD Confusion

No matter how much cooperation lies behind the creation of a DTD, any large enterprise will be faced with multiple XML standards. In some cases, a single problem domain may have multiple standards supporting it. In other cases, a business may have needs

that span multiple problem domains. The result, however, is a new integration problem — the problem of integrating diverse XML standards.

XSL again comes to the rescue in solving the problem of DTD proliferation. Just as XSL can be used to translate one XML standard into HTML, it can also be used to translate it into any other XML dialect. (See sidebar "The Display of XML.") Figure 7 shows the integration of systems for two different businesses using two different XML standards.

Again, the first business exports its shared data via a Java servlet in accordance with the XML standard chosen by that business. At the other end, the business partner reads the XML via the Web and uses the XSL transformation to convert it into the XML standard chosen by the business partner. It is then able to use that data as it would any other data set in the XML standard it understands.

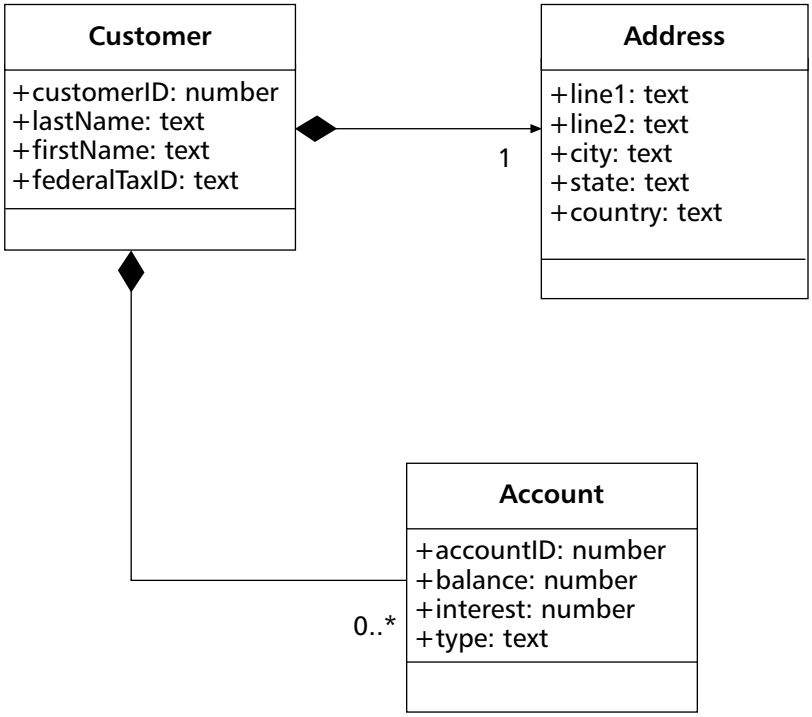

Figure 6 — A hypothetical banking industry XML standard.

BizTalk

To take advantage of XSLT to mitigate the proliferation of XML standards, an XSL translation must exist specifically for the combination of standards in question. If no such translation exists, then a business will have to write its own XSL. In an industry where many standards dominate, this problem could get out of control.

Microsoft and some of its partners are promoting a solution to this problem called BizTalk. The goal is to anoint certain XML standards as BizTalk compliant, thereby guaranteeing that a translation exists between it and other BizTalk standards. The ability of the BizTalk initiative to succeed, however, is uncertain. It may be too much to expect every DTD author to define XSLT translations for their standard into every other relevant standard. More information on BizTalk can be found at www.biztalk.org.

Custom Standards

One thing a business definitely should not do is wait on others to

THE DISPLAY OF XML

A good XML dialect contains no style information. In other words, you cannot tell from an XML document (except, of course, HTML documents) how it should be displayed. You will likely want to display a single XML document in many different environments (e.g., the Web and printed documents). It would be especially odd if you could not display XML on the Web, since the Web is what drove the creation of XML. Display is the problem that XSL helps to solve.

To display XML content on the Web, XSL translation to HTML is required. An XSL stylesheet is used to define how an XML document should be displayed by converting the industry-specific XML into HTML tags that can be understood by a Web browser.

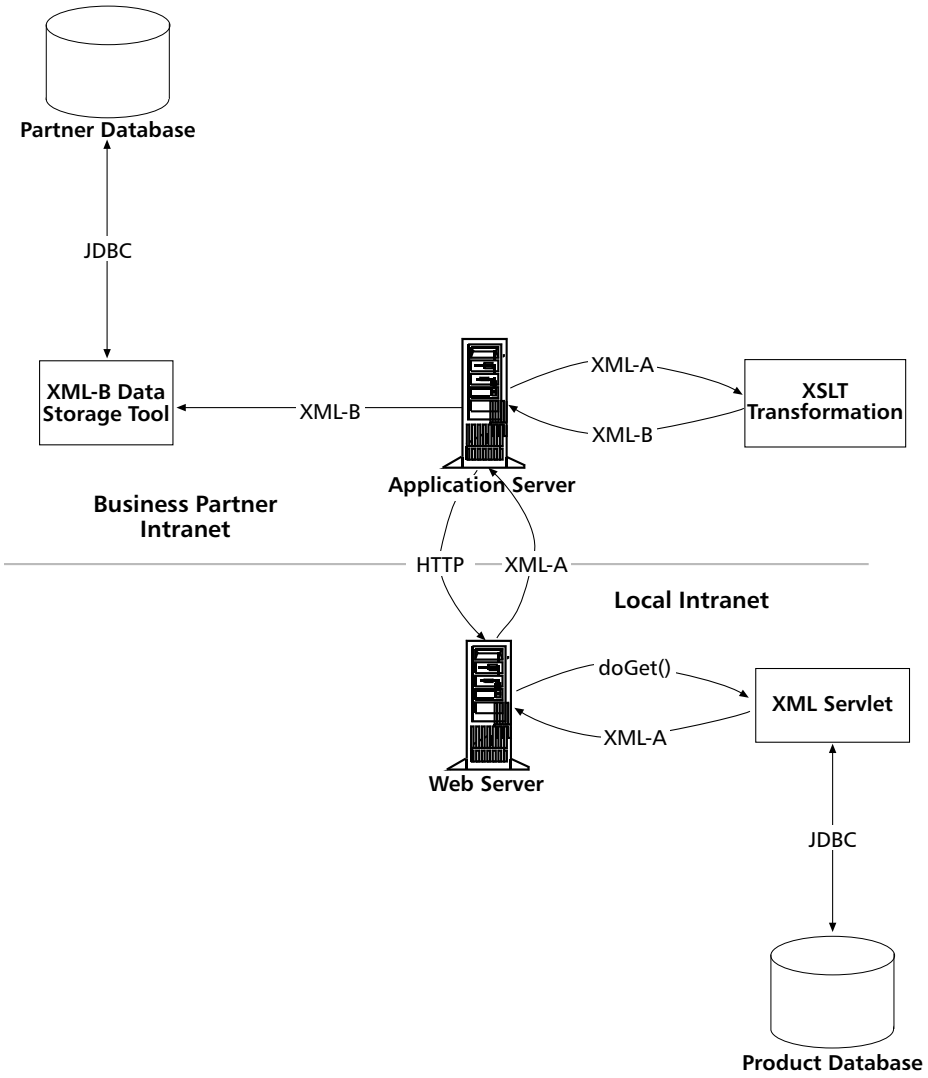

Figure 7 — Integrating businesses that have two different XML standards.

```
<?xml version="1.0"?>
<document id="ger20000201">
   <title>The Problem Domains of XML</title>
   <section>
      <heading>Introduction</heading>
      <paragraph>
        Important technologies inspire innovation.
      </paragraph>
   </section>
</document>
```

Figure 8 — Example of source code for a document written in XML.

move forward with XML. Nothing stops a business from defining a custom DTD to meet its business needs. The caveat, of course, is that no one else will understand that dialect of XML. If an industry standard does eventually evolve, however, XSLT will easily solve the problem of migrating to that standard.

STRUCTURED DOCUMENTS

Integration is not the only problem domain for XML. The creation of structured documents is another important, if less pervasive, problem domain. Earlier, we discussed the problem of college textbooks. The XML solution is to define a DTD for the representation of structured documents.

Figure 8 contains the source code to a document written in a mythical structured document XML vocabulary. Such a vocabulary might contain some minimal style information such as bold and italic tags. In general, however, it would break a document down into the structural components common to documents. When the need arises to display that document, either for the Web or for publication, the display tool can determine how to display the document based on its structure.

This document can then be printed on its own, in a magazine, or as part of a book. Thus, if you want to create a custom book from chapters in existing books, you can use the XML to

reference external documents and construct the book using a few simple <chapter> tags. Figure 9 illustrates how the custom textbook can be created in XML simply by pointing to chapters in existing books.

XML's ability to provide structure to documents is just one part of its ability to solve this problem. The combination of XLink and XPointer is another critical element in building custom texts. It is wasteful to copy the XML from the chapters of the original documents and build a new XML document. Instead, XLink and XPointer enable a DTD to define the <chapter> tag as a linking tag that will instruct interpreters of the document to go out to the original textbook to grab the text. The professor could thus save the XML on the college's system without compromising the intellectual property of the publisher. In addition, whenever the professor requests the custom text to be printed, the text will contain the most recent versions of those chapters, even though no one has touched the contents of the custom XML book.

Being able to decompose and recompose a document is just one of the benefits of using a structured document format; another benefit is format independence. Although most document authoring tools enable the conversion of a document among most formats, they invariably lose something in the translation.

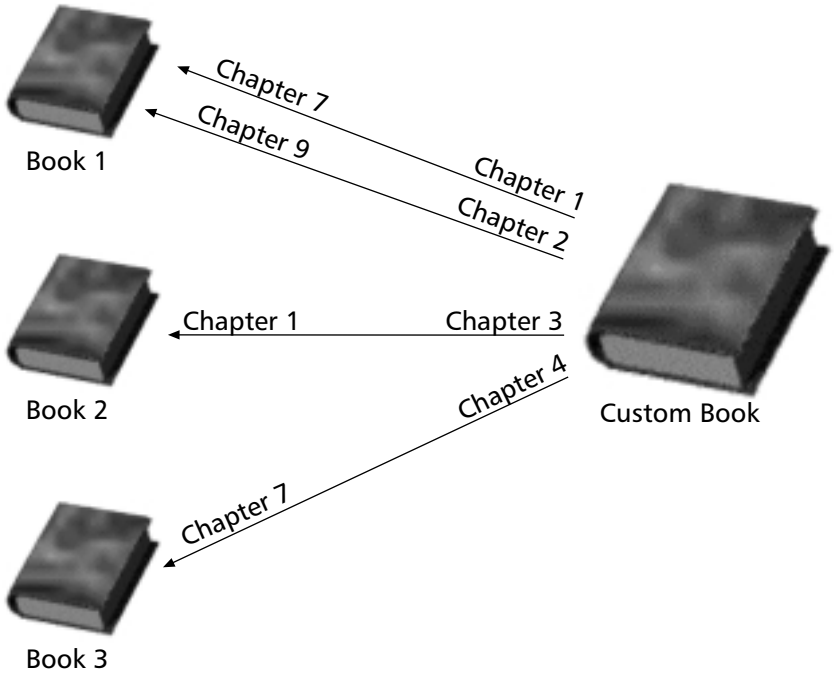

Figure 9 — Creating a custom textbook using XML pointers.

This problem is due in part to the fact that the native formats of most document-authoring tools are difficult-to-parse binary formats or very terse markup languages. Whether binary or text, they are almost always proprietary in nature. The style information behind one set of tags in one vendor's document format may not have an exact translation in another vendor's document format. Storing data in an open, structured format makes it much easier to translate into stylized formats based on the contents of the document.

AN XML CASE STUDY: DISTRIBUTION CHANNELS

Figure 10 illustrates the flow of information through a hypothetical distributor, Collegiate Distributors. The business and the industry have been changed for this example, but they are based on a real-world set of business problems. The problem for Collegiate is that the Internet is changing the company's business model. The steady growth Collegiate has experienced over the past 50 years has suddenly accelerated — mostly by accident — due to its first forays into e-commerce.

Collegiate's business involves the publication of catalogs aimed at colleges and universities. These catalogs contain educational texts the company distributes, along with reviews that help professors evaluate the texts as material for their courses. The value Collegiate provides comes in the form of these reviews written by the company's highly specialized staff. For

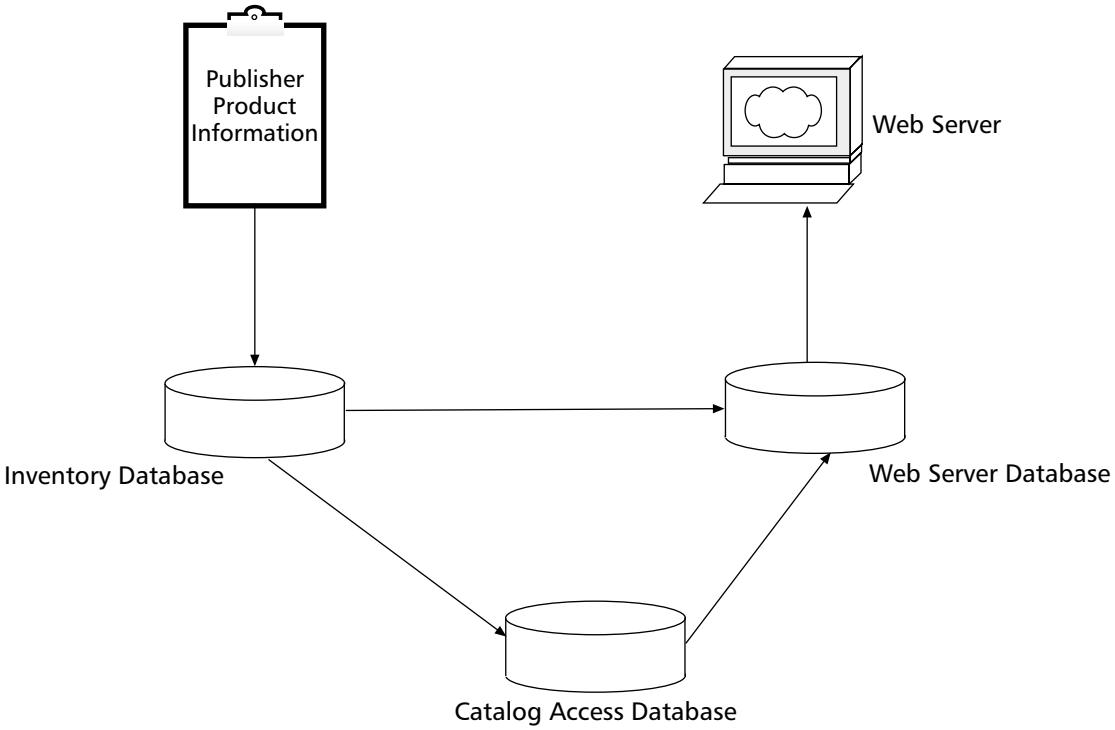

Figure 10 — Information flow through Collegiate.

the past 50 years, its business has involved receiving product information from publishers, reviewing the products, and producing a catalog containing those reviews. Collegiate then mailed the catalogs to its target colleges and universities and fulfilled the orders placed by college bookstores.

The business's growth was slow and steady until it decided to put its catalog on the Web and begin taking orders directly from individuals. This decision was made more because Collegiate had a Web site and was trying to figure out what to do with it than because the company had any sort of strategic vision behind the e-commerce attempt. Nevertheless, the Web site proved extremely successful, and the company is faced with two problems. The first problem is that its antiquated systems are unable to deal with the sudden activity. The other problem is that Collegiate needs to learn how to handle the sudden growth and develop a strategic plan to move the company into the future.

The Business Problem

Obviously, Collegiate cannot create a technology solution until it has a strategic business vision for the new technology. The question before the company is whether to embrace the recent change or abandon the e-commerce effort and return to the old way of doing business.

Collegiate has a luxury most businesses faced with the same question do not have. Namely, it can go back to the old way of doing business, at least for the short term. As a privately held company, it is not required to seek the sort of growth it has seen over the past year. Furthermore, because the new business is not part of its core competency — i.e., direct selling instead of distribution — it is not faced with the issue of accepting this new business model or being obsolete.

Nevertheless, money talks, and the company has made the decision to embrace e-commerce with the goal of building an IT infrastructure that will make it possible for Collegiate to more

quickly adapt to changing conditions in its business. Specifically, Collegiate sees the following three main areas in which a modern IT infrastructure can help it embrace the e-commerce business model:

- The current system is a set of hacked-together Perl scripts that pull the data from the published catalog and throw it up on the Web. A modernized Web site could be more tightly integrated with the company's other systems and thus be able to support more business than simply taking orders for products. Furthermore, with more intelligence behind the catalog information, the company could target products based on visitor profile information.
- The current catalog-building process is extremely time consuming, resulting in one major catalog per year and four minor quarterly catalogs. With a more modern IT infrastructure, Collegiate envisions the ability to build catalogs on demand targeted to specific campuses. By combining a modern catalog building process with a new Web site, the targeted content can come in the form of custom catalogs.
- By making all of the IT systems simpler to integrate with external systems, Collegiate can more easily engage in strategic partnerships with publishers and universities.

The Technology Problems

Figure 11 shows the existing IT infrastructure for Collegiate. It is a hodgepodge of antiquated technologies that worked well for Collegiate's old business model. The center of the business is an AS/400 containing Collegiate's product information. Just about all business processes supported by IT run through the inventory system. The other major system is the recently built e-commerce system made up of Perl CGI scripts.

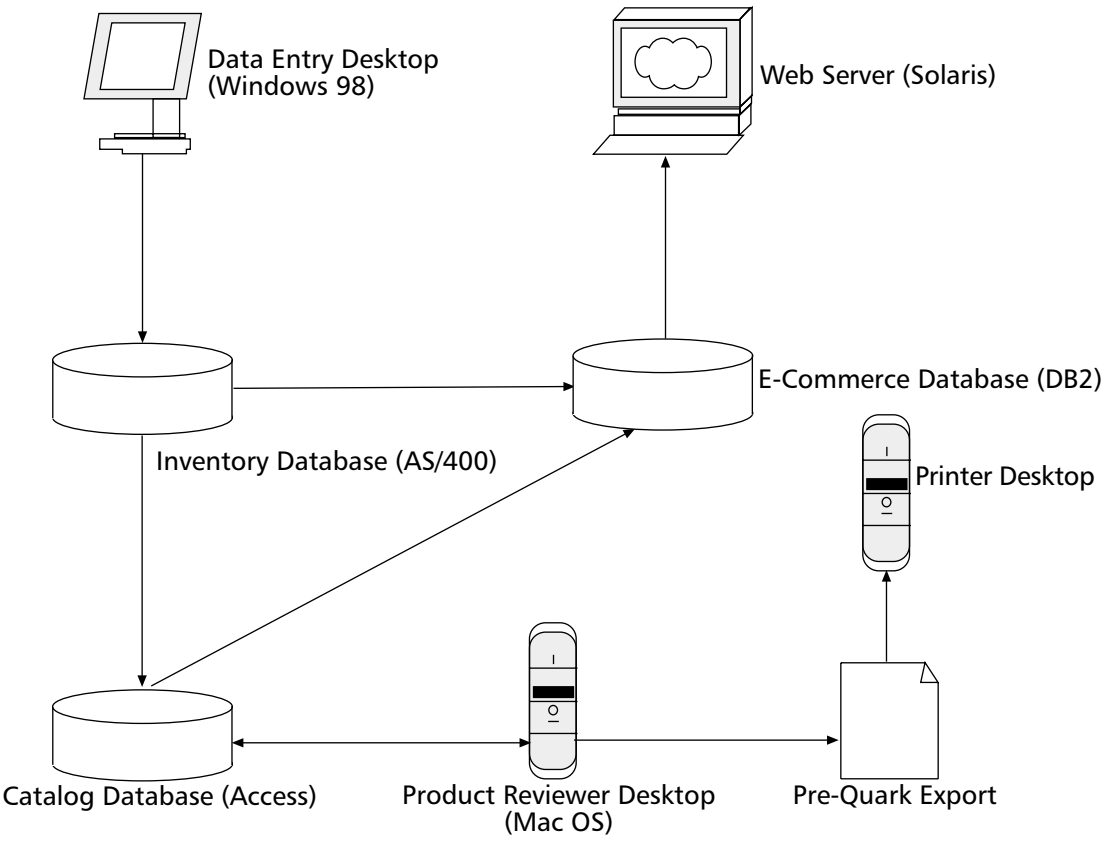

Figure 11 — Collegiate's existing IT infrastructure.

The Inventory System

The inventory system is an AS/400 system with very little thought put into the data model supporting Collegiate's products. As a result, the structure of the data is specific to Collegiate's current business of distributing books. If the company moved to selling other sorts of educational products, its AS/400 system would have to be twisted into supporting the new product set. As long as it continues selling books, however, the AS/400 system will continue to work. Replacing it is therefore not currently a critical-path priority according to the stated business goals.

The Catalog System

The catalog-building process is not really supported by the IT infrastructure. A few database tables exist in an Access database to store the reviews, but no application ties the catalog-creation workflow together. The process of building a catalog is thus a manual process that can take several months. In general, it requires the following steps:

1. The reviewers — people whose expertise is in understanding the subject matter in educational texts, not database querying — spend a great deal of time working out the right queries for the AS/400 and Access databases to support sections of the catalog they are trying to build.
2. Using a text editor, the reviewers write up their reviews and cut and paste them into the Access database. During the process of writing these reviews, they sometimes need to query across the AS/400 and Access databases (a two-step process since there is no integration between the two).
3. The reviews in Access are copied into Quark XPress and saved as a Quark document.
4. The Quark reviews are merged together, requiring a manual edit of the internal Quark formatting.
5. The resulting merged document is sent to the printer for layout and printing as a catalog.

The key to being able to meet the stated business goals is to support the workflow behind catalog building. It is also important to get reviewers focused on their core competency: reviewing textbooks. By removing the requirements that reviewers be novice technologists, Collegiate can reduce the training required to make new reviewers productive.

The E-Commerce System

The other major system at Collegiate is the new Web system. Although it is new, it was not well planned. The Perl CGI scripts are slow, cannot handle much more traffic than they currently handle, and do not perform tasks such as profiling that are critical to the company's business goals. In addition, only one person in the company actually knows Perl. Losing that developer could cost the company in terms of lost sales and lost mind share. The company therefore needs an enterprise-class Web infrastructure that integrates the Web site with both the forthcoming catalog system and the old inventory system.

E-Business Integration

Collegiate receives information about its publishers' offerings in a variety of formats that are largely dependent on the level of maturity of the individual publishers. No matter how mature those publishers are, however, Collegiate ends up manually entering all of that data into the inventory system due to the lack of sophistication in its own systems, leading to a rather large time lag between when a book is published and when it is available for review for a catalog. To increase the efficiency with which Collegiate integrates with its publishers, it needs a strategy for electronically pulling in product information from publishers.

XML for the Catalog System

The catalog system exemplifies two major problem domains, identified earlier in this report, that fit XML's strengths. First, the book reviews are examples of structured documents. The ability to review books and then combine and recombine those reviews on a whim for publication both in print catalogs and on the Web can be facilitated by Collegiate's storing the reviews as XML documents. Second, the catalog system needs

to be able to import data from the legacy inventory system and export it to the e-commerce system. A custom XML dialect would be ideal for solving this problem.

Reviews as Structured Content

As we discussed earlier, the current system stores reviews in an Access database as plain text. The reviewer later copies each review into Quark and adds style to it. The style information is completely lost to the Web system, and the process for combining the reviews into a catalog is very labor intensive. The catalog system therefore needs an XML dialect that will enable reviewers to not only use a WYSIWYG editor to add style information but also store information about the review and its relationships to Collegiate products. Figure 12 shows some potential XML that might support a Collegiate review.

Like HTML, XML enables authors to embed links to external documents. In the case of Collegiate's review XML, the link information can be embedded in a `<citation>` tag so that a translation to HTML is adaptable enough to enable a user to see the review of the cited book while at the same time formatting book titles in italics for printing in the published catalog. With the review stored in XML, a single XML-to-Quark translator can be written that will stylize the review based on its structure. The result would be an entry in the catalog that looks like Figure 13.

```
<?xml version="1.0"?>
<review id="7293774">
   <title>The History of Art</title>
   <asset id="320988792" type="product">
      <price>$70.00</price>
      <property>
        <propName>year</propName>
        <propValue>1990</propValue>
      </property>
      <property>
        <propName>ISBN</propName>
        <propValue>XXXXXXXX</propValue>
      </property>
   </asset>
   <rating>7</rating>
   <text>
      <citation>The History of Art</citation> is
      an old, yet still relevant coverage of art
      history for introductory art courses.  It is
      not recommended however for courses geared at
      art majors.  For introductory courses aimed at
      majors, we recommend <citation xml:link="simple"
      href="http://catalog/2988303">Art in Focus
      </citation>
   </text>
</review>
```

Figure 12 — A Collegiate XML book review.

The History of Art (1990) — ★★★★★★★

The History of Art is an old, yet still relevant coverage of art history for introductory art history courses. It is not recommended for courses geared at art majors. For introductory courses aimed at majors, we recommend *Art in Focus*.

ISBN: XXXXXXXX

Your price: $70.00

Figure 13 — A sample catalog entry.

Because the full structure of the document is now stored in XML, catalogs can be constructed on demand, simply by querying the AS/400 system for products that meet whatever criteria make sense at the time. The only requirement is that the product have been reviewed previously.

One part of the XML document that might initially appear odd is that the product is referenced using an `<asset>` tag, not a `<product>` tag. This oddity is a simple attempt to decouple reviews from specific products. After all, as the business evolves, Collegiate may want catalog entries about things that are not books — that perhaps are not even products. This XML dialect views reviews as digital assets that have some sort of physical reference asset.

Integrating the Catalog System

The second application of XML to the catalog system is integrating it with the other systems at Collegiate. The inventory system is still the system of record for product information. The catalog system therefore needs a mechanism for pulling product data from the inventory system and populating the catalog system. It also needs a tool for exporting reviews to the e-commerce system.

Because Collegiate has already defined an XML dialect for structuring its reviews, it has already handled the problem of how to export the reviews in the catalog system to the Web. This benefit illustrates one reason why XML is such a powerful tool for these types of business problems. Specifically, it is likely that many different systems will require a structured view of business data. The old approach might have been to write custom interfaces among the different systems that suited the individual needs of each integration effort. The XML approach captures the business concept (in this case, a book review) in a structured format. Any system needing information about that business concept can standardize on that one structured format and tailor the data to its needs.

In the case of the e-commerce system, what it really wants from the catalog system is simply the ability to retrieve reviews. Much of the other data used by the Quark export is irrelevant — it gets that information directly from its interface to the inventory system. It cares only about being able to relate reviews to products from the inventory system.

Of course, the catalog system gets all of its product information from the inventory system. It needs the inventory information required to perform the queries upon which reviewers construct a catalog and nothing more. For Collegiate, this information is the price, sales ranking, and source area. The inventory system, however, does not care about the specific needs of the catalog system. Collegiate needs to take care not to think too narrowly and define an XML dialect for this interface that suits only the needs of the inventory-to-catalog system integration.

Collegiate first looked toward an industry standard for capturing product information as its lingua franca for exporting inventory information. Finding no standard that suited its needs, Collegiate defined its own XML dialect. Figure 14 is the XML document that exposes the *History of Art* product to the world, including the catalog and Web systems.

Products are only part of the inventory system's information. It also must be able to export other data such as manufacturer information. The catalog system, however, is only interested in the product information. The inventory import reads these XML documents and populates the catalog database with the information the catalog system cares about. The application supporting the business processes of building a catalog can then use that information to enable reviewers to review products and assemble catalogs. Finally, the Web system and Quark export tool are both able to access review information via the review XML.

The challenges facing Collegiate are much greater than a simple application of XML can fix. It still needs a solid architecture that can support its new e-commerce system and enable the functionality

e-business application delivery

Extensible Markup Language

Run for the hills! Any time IBM, Sun, Microsoft, Oracle, and Hewlett-Packard (and nearly any other technology supplier you can mention) all agree on something, it is time to check out the "hype indicator." Extensible Markup Language, or XML, may be the only technology in recent years to garner such universal acclaim. It is refreshing to note, however, that IBM is sniping at Microsoft's XML implementation; after all, if the technology was going too smoothly, there would be even more cause for alarm.

There are five basic questions about XML to be answered: What is XML? What are the driving economic forces behind XML? What are the challenges in using it? What are the uses for XML? What should you be doing now?

With all the hype about XML, one might think it was a new technology — some grand new scheme that no one had ever considered before. In fact, it's really part of an ongoing evolution. The most frequently cited charac-

executive summary

"XML is the millennium elixir," joked Bryan Caporlette, vice president of Sequoia Software. The Extensible Markup Language (XML) radiates importance. Java garnered wide support (except from you-know-who) because it was touted as a universal language for interoperability. But Java left out one piece — actually a very, very big piece: data. XML may fill in this data gap. And then, Java, the universal programming language, and XML, the universal data language, go riding off into the sunset together.

However, as we all know, reality always catches up with us, and it never really matches the ending of the movie. XML's importance derives not only from what it is, but also from what it does — it reemphasizes the critical importance of data in our e-business world. This issue of ***eAD*** explores XML and its impact on application integration and e-business.

Jim Highsmith, Editor
E-mail: jimh@adaptivesd.com
Internet: www.adaptivesd.com

january 2000 vol. XII, no. 1

CUTTER CONSORTIUM

that its business goals demand. This includes questions like what sort of HTML generation technology (Active Server Pages, Java servlets, etc.) to use, what sort of distributed computing technology, if any, to build the e-commerce system with, and even whether or not Collegiate should buy an off-the-shelf e-commerce solution. XML can solve none of those problems. As this case study has shown, however, XML does solve some issues critical to Collegiate's strategic vision very well.

```
<?xml version="1.0"?>
<product id="320988792" type="book">
   <name>The History of Art</name>
   <manufacturer>282743393</manufacturer>
   <price>$70.00</price>
   <inventory>124</inventory>
   <salesRank year=1998>301</salesRank>
   <salesRank year=1999>324</salesRank>
   <book>
      <discipline primary="true">art</discipline>
      <discipline>history</discipline>
      <discipline>humanities</discipline>
      <year>1990</year>
      <isbn>XXXXXXXX</isbn>
   </book>
   <related>24320002</related>
</product>
```

Figure 14 — A product for the inventory system.

XML CHALLENGES

XML does not come without its share of pitfalls. Most of these issues are challenges common to any new technology: marketing hype, the lack of tool support, and the immaturity of the technology. This report has already addressed one downside of the hype that surrounds XML: the difficulty in assessing what problems XML addresses well. There is another downside to the hype — marketing departments latch on to it and attach XML as a bullet point to just about every enterprise product. As a result, identifying exactly what a vendor means when it claims its product is XML-enabled can be difficult. Does that mean the vendor has created some custom XML DTD and saves all of its information in this dialect? Or does it mean that the product understands industry-standard XML dialects related to its problem domain? IT decisionmakers will have to carefully question vendors as to what they mean when they put XML on their checklists of supported features.

As with any new technology, the tools to support XML in the enterprise are immature. The growing use of XML is highlighting some very key tools that are either missing or incomplete at this time. The largest hole lies in authoring tools. Part of this problem arises because the formalization of different XML vocabularies either has not happened or the ink has not yet dried on the specification. It is difficult to build an authoring product aimed at nonexistent specifications; thus, at this time, authoring XML documents is almost always a rather manual process.

Also needed is a class of tools to support the DTD design and development effort. Earlier in this report, UML was put forth as a good standard for expressing the design of an XML vocabulary. I am currently aware of no tools, however, that support modeling a problem domain in UML with the goal of producing an XML DTD. Even if you do not accept this report's approach of using object-oriented software engineering practices and UML in the design of XML standards, a business could benefit from a tool that supports the DTD design process.

Finally, and perhaps most importantly, most enterprise applications and tools are not yet truly XML-enabled. Your database engine, for example, probably does not have a tool for mapping your data model to an XML standard appropriate for your industry. Your EJB application server probably does not have a tool for exporting your beans as XML documents. And your financial application probably does not know how to read in imported data using a financial XML standard

like FinXML. Ultimately, these tools will come, and XML will be integrated into nearly every enterprise-oriented application. Today, however, most of the work of using XML in a business will be up to the IT department.

We have already discussed the problem of standards — namely that a standard that addresses your industry may not yet exist or may not yet be stable. The fact is, XML itself is based on a host of standards (XML, DTD, XSL, SGML), of which only SGML is a stable standard. XSL in particular is a very fast-moving target. Only a year ago, XSL was nothing more than an XML-specific stylesheet language competing against HTML's Cascading Style Sheets. It has changed a lot in the past year, and, as we have discussed, now supports XSLT transformations between XML dialects. Although this cast of characters appears to be stabilizing, newer support specifications are appearing on the horizon and moving quickly. Among the stars of these specifications are XPath and XML Namespaces.

Another problem related to the newness of XML is the lack of technologists with XML in their toolbox. This problem is one that is hitting the Java community fairly hard. XML should not prove as problematic for IT shops as Java was, since its syntax is familiar to anyone who knows HTML and it is, after all, a markup language and not a programming language. Unlike most markup languages, however, XML will require an IT shop to have good business analysts and object-oriented designers if the IT shop intends to engage in designing DTDs. These people are hard to find across the board; finding ones with knowledge of XML and the issues surrounding DTD definition is even more difficult.

THE FUTURE OF XML

The year 2000 is certain to be the year in which XML becomes the preferred method for data interchange. Although it was not the original focus of the XML hype, the data interchange problem has become the feather in the XML cap. This *Executive Report* has hopefully made it clear exactly why that is. As a result, most businesses will be looking forward to the development of XML standards that address their individual industries. The page www.xml.org/xmlorg_catalog.htm provides a good look at some of the standards being developed. There are many other exciting developments you should be watching for in the XML world as well.

As we discussed in the previous section, tools that support XML are lacking. In 2000, the industry should see vendors seriously incorporating XML into their products. Most of the large players, including Oracle, IBM, Microsoft, Sun, and others, have already committed to support for XML across the board. Internet Explorer already has XML support. Java has XML APIs. IBM has developed a very popular XML parser called SAX. Going forward, all of these companies plan to integrate XML across their product lines.

A more mundane development should be the growing acceptance of HTML 4.0 as a standard for rendering Web pages. This development is important because HTML 4.0 (also known as XHTML) is an XML dialect. As such, it is bound to much stricter rules regarding the authoring of proper HTML pages. Right now, the Web is filled with badly written HTML documents. HTML browsers are extremely difficult to write because they must be capable of dealing with the forgiving nature of non-XML HTML, as well as incorrect HTML.

The advantage of the adoption of HTML 4.0, however, is not in making life easier on browser writers. HTML 4.0 makes the maintenance of HTML documents much simpler. As things stand today, revamping a Web site is an extremely painful process, largely because HTML 3.2 is burdened with years of forced acceptance of proprietary standards; bad authoring habits compound that mess. HTML 4.0 imposes the structure of XML authors and moves style information into stylesheets. As a result, a company can easily make changes to the look of a Web site without affecting its content.

XML support in different programming languages is going to be key to people's ability to deliver XML solutions. Many languages already have some sort of XML bindings. The most exciting work appears to be occurring in Java. In 2000, XML-oriented technologists should be looking for the stabilization of XML APIs in their language of choice. Another related development in XML to be monitored is support for the Document Object Model (DOM) in tools and APIs. DOM is a language and platform-independent standard for representing the structure of a document so that programmers can manipulate the document programmatically. Projects that need to manipulate XML documents will find it advantageous to follow DOM, as it helps abstract away from the particulars of a given XML document. The result will be tool components that can be reused in new tools.

One important area to watch is the development of an XML query language. Various proposals for such a beast have been thrown about since XML's beginnings, but the acceptance of a standard has been slow. The point of an XML query language is specifically to do for XML documents what SQL does for relational databases — that is, enable people to query against XML documents. Two major contenders are emerging for a query language: XQL and XML-QL.

On the cutting edge of XML is the development of supporting standards like XPath and XML Namespaces. XPath is a language for addressing parts of a document. XML Namespaces is a standard for delimiting namespaces for XML elements so that a system using two different DTDs with an `<event>` tag does not see namespace collisions. IT departments that have committed to XML will want to watch for the appearance of new standards like these and monitor their development.

SUMMARY

Technologies that enhance the points of contact among business entities have the power to redefine business models. The Internet is a strong example of this fact. It has fundamentally changed the way in which a business interacts with all of its business partners at all levels, from customers and clients to vendors and employees. Businesses that have embraced the Internet have discovered a powerful competitive advantage. This advantage is so strong that it is now becoming the norm, leaving businesses that have avoided the Internet scrambling to remain relevant.

XML is a new point of contact in business. The adoption of XML for enterprise systems integration greatly increases the efficiency of business processes at the point of integration with business partners, especially on the Internet. Because of these increased efficiencies, the demands of companies to integrate with their partners will only increase. As a result, those who ignore XML will likely find themselves in the same boat as those who have ignored the Internet. They will be cut off from the outside world and fighting to get back into the game.

No technology solution is a panacea, and XML is no different, despite rabid XML evangelists' claims to the contrary. This report has shown how it is ideally suited to the problems of systems integration and structured document management. Its success in those realms is related directly to its core competency: managing structured content.

Certainly, XML will prove to be useful beyond the problem domains covered in this report. The challenge thus remains to wade through the hype to identify those problems where a solution in which structured content can help. In those situations, XML is an excellent candidate to solve the problem.

ABOUT THE AUTHOR

George Reese is the chief technology officer for Ancept, Inc. in Minneapolis, Minnesota, USA. He has been a pioneer in Java enterprise development since creating the first JDBC driver in early 1996. Among his publications are *Database Programming with JDBC and Java* and *MySQL and mSQL* (O'Reilly). His focus at Ancept is enterprise Java architectures with digital asset management components. He can be reached at george.reese@excite.com.

ACCESS TO THE EXPERTS

TRY OUR E-MAIL ADVISORS — FREE!

Sign up today and get a FREE trial subscription to any of our E-Mail Advisors

- Business-IT Strategies
- Trends
- Sourcing
- Architecture/ e-Business
- The Cutter Edge
- Research Briefs

Register at our Web site:

www.cutter.com/consortium/

CUTTER CONSORTIUM

MAXIMIZE YOUR IT BENEFITS

Maximizing the business value of IT continues to rank at the top of CIO and CEO priorities.

As an IT professional, your goal is to provide cost-effective information technology that is perfectly consistent with the strategic, tactical, and operational business objectives of your enterprise.

It needn't be a pipe dream.

Chances are, your company is spending an enormous amount of effort and money on IT. And chances are, management questions the business value of that investment.

Cutter Consortium provides a variety of products and services that will help you optimize your IT investments, including:

- Business Intelligence Advisor
- Business-IT Strategies
- Business Technology Trends and Impacts
- Component Development Strategies
- Cutter IT Journal and CIT E-Mail Advisor
- Distributed Computing Architecture/e-Business
- e-business Application Delivery
- IT Metrics Strategies
- Sourcing
- Consulting, Evaluations, Assessments, and Training
- Custom Benchmarking and Research Reports
- Online Resource Centers

For information about any of these services, or to set up a demonstration for your company, contact us by phone, fax, or e-mail:

Margaret Rogers
+1 781 648 8702
+1 781 648 8707 fax
mrogers@cutter.com

Dennis Crowley
+1 781 641 5125
+1 781 648 8707 fax
dcrowley@cutter.com

Rick Doody
+1 781 641 5104
+1 781 648 8707 fax
rdoody@cutter.com

Cutter Consortium Distributed Computing Architecture/e-Business Advisory Service

Senior Consultants

Douglas Barry

on storing objects using DBMSs

Douglas Barry is principal of Barry & Associates, Inc., which he founded in 1992, and executive director the Object Data Management Group, an industry standards organization. Mr. Barry has worked with DBMS technology for more than 20 years and with object technology and databases since 1987. He has been actively involved in creating and promoting standards for storing objects in databases. Mr. Barry specializes in the strategies, technologies, and products associated with objects and object-relational database applications.

Peng Boey

on component-based architecture for e-business applications

Peng Boey is vice president of Consulting Services with NetNumina Solutions. With more than 10 years of IT experience, Mr. Boey is an expert in distributed systems architecture. He has provided professional advice to Global 1000 companies on how to design, build, and deploy component-based architectures for e-business applications. He has been a leader in creating the VIEW methodology, a revolutionary architecture process that utilizes the latest enabling technologies for building mission-critical e-business systems for the Internet, as well as for intranets and extranets. Currently, Mr. Boey is conducting research on developing component frameworks for rapidly building e-business solutions.

Theodore R. Burghart

on developing heterogeneous client-server and distributed systems

Theodore Burghart is principal engineer at Quoin, Inc. He has extensive experience with heterogeneous client-server and distributed systems design and development. His projects have included communications, database, technical, and process control services implementations. Mr. Burghart is experienced in cross-platform enabling technologies, such as CORBA as well as with LDAP, and with relational, object/relational, object-oriented, and full-text databases. Currently, Mr. Burghart collaborates with development teams to define and construct CORBA-based infrastructures. In addition, Mr. Burghart provides technology training and consulting services to clients in the healthcare, insurance, and financial services industries.

Richard Dué

on component development methods and project management

Richard T. Dué is president of Thomsen Dué and Associates Limited. He specializes in object and component development methods and in object technology project management. Mr. Dué has developed and presented information technology training courses in 28 countries to participants from hundreds of organizations. He is a member of the OPEN methodology consortium and has been actively involved in developing business object standards. Mr. Dué is a frequent contributor to the *Cutter IT Journal*, and has held various management positions in the public and private sector in the US and Canada.

David Frankel

on Java- and Internet-based component architectures

David Frankel is chief scientist at Genesis Development. He assists clients in developing and customizing advanced component architectures based on CORBA, DCOM, Java, the Internet, and related technologies. Mr. Frankel has been instrumental in formalizing advanced component architecture to support large-scale software development and systems integration. He is a member of the OMG Architecture Board, was a major contributor to the CORBA/COM Internetworking standard, and is cochair of the OMG Business Object Initiative Working Group.

Max Grasso

on distributed secure transaction systems

Max P. Grasso is chief technology officer of NetNumina Solutions. He is a recognized expert on distributed secure transaction systems with a focus on high reliability, mission-critical applications. He also has significant expertise in the management issues involved with deploying such systems. Mr. Grasso has been at the forefront of distributed computing technology since its beginning, both as a member of the Open Software Foundation's team and as a cofounder of the Open Environment Corporation. As CTO of Internet Business Solutions, his mission was building the technology for the execution of secure distributed transactions on the Internet. In that role he designed a framework for business-to-business transactions and interenterprise transactional workflows. Mr. Grasso has overseen the architecture and the design of large systems in the telecommunication, financial, banking and gaming industries.

Michael Guttman

on transitioning to enterprise component technology

Michael Guttman is chief technical officer and cofounder of Genesis Development, where he assists clients in planning their transitions to enterprise component technology. Mr. Guttman, who has been a pioneer in the use of component and object technology for large-scale distributed systems, is a specialist in advanced component architectures. Mr. Guttman has more than 20 years of experience in software development and has been a major contributor to several OMG standards, including CORBA 1.0, CORBA IIOP, and CORBA/COM Internetworking.

Curt Hall

on data warehousing and data management strategies

Curt Hall, editor of *Business Intelligence Advisor*, is an expert on data warehousing technologies and products. His

SENIOR CONSULTANTS

recent study on the corporate use of data warehouses and the issues associated with data warehousing projects has resulted in the in-depth report *Data Warehousing for Business Intelligence*. Mr. Hall is the coauthor of *Intelligent Software Systems Development* and a contributing editor to James Martin and James Odell's *Object-Oriented Methods: Pragmatic Considerations*. He is the former associate editor of *Object-Oriented Strategies* and *Application Development Strategies*. Mr. Hall's work has appeared in technical journals such as *IEEE Expert*. He has also been an organizer of and speaker at industry events such as *ObjectWorld*.

PAUL HARMON

on distributed computing and component development for business applications

Paul Harmon is a well-known consultant and analyst of software trends. Mr. Harmon has been very influential in the movement to commercialize object and component technologies for business applications. Mr. Harmon recently completed a study of the acceptance of object technology and components in corporate development groups. As editor since 1991 of *Component Development Strategies*, published by Cutter Information Corp., Mr. Harmon has studied the commercial and business applications of object technology. He has also been the editor of three other Cutter Information Corp. newsletters over the years: *Intelligent Software Strategies*, *Application Development Strategies*, and *Business Process Strategies*. Mr. Harmon is a frequent speaker on the strategic impact of new software technologies on business. Mr. Harmon is the coauthor of several books.

IAN HAYES

on e-business strategy

Ian Hayes is president of Clarity Consulting, Inc. where he provides management consulting on IT strategies, emerging trends, markets, and challenges. Mr. Hayes has advised dozens of *Fortune* 1000 companies on many IT issues including outsourcing, process redesign, e-business strategies, efficiency enhancement, productivity and service level metrics, service offering development, and product and service positioning. A frequent and popular speaker on a variety of IT topics, Mr. Hayes is also a regular contributor to Cutter Information Corp.'s *Cutter IT Journal*, writes the Managing e-Business column for *Software Magazine*, and is on the editorial advisory board of the *Enterprise Application Integration Journal*. He has also written dozens of articles and white papers, and coauthored two IT books. Mr. Hayes cofounded Language Technology, Inc., an early software redevelopment product vendor, and served as a manager at Keane, Inc. before founding Clarity Consulting in 1993.

J. BRADFORD KAIN

on distributed business components

Brad Kain is CEO and cofounder of Quoin, Inc., providing consulting, mentoring, and software development services in object and distributed technology. Mr. Kain has used object-oriented analysis and design since 1987. He has helped define the use of object and distributed technology to realize distributed business components. This work has involved the specification of sophisticated intranet, Java, and distributed applications. Mr. Kain has managed the technical direction and development teams of distributed application infrastructure development projects for managed care, client management, general ledger, securities trading, marketing, engineering and manufacturing design, and other applications. Mr. Kain has participated in the work of the Object Management Group's Technical Committee on CORBA and the specification of domain services.

ANDRÉ LECLERC

on formal specification approaches to the development of information and management systems

André Leclerc is the director of development for Technology Development Associates, Inc., where he is active in developing, training, consulting and mentoring object-oriented information systems. Mr. Leclerc's interest is in formal specification approaches to the development of information and management systems. In 1984, Mr. Leclerc was appointed vice president of Yourdon, Inc. Following his tenure at Yourdon, Inc., he served as vice president of Kenneth G. Moore and Associates. Mr. Leclerc has authored a book on structured PL/1, and a variety of articles, seminars, and tutorials on information systems, including the OO seminars for Ptech, Inc.

JEAN PIERRE LEJACQ

on architecture and implementation of distributed systems

Jean Pierre LeJacq, an experienced architect, designer, and implementer of distributed systems, is CTO and cofounder of Quoin, Inc., providing consulting in object and distributed technologies to clients worldwide. Mr. LeJacq is the architect and technical lead for the development of an infrastructure for distributed application development, and is responsible for the design and implementation of a CORBA-based system. He has extensive experience in Java, C++, and UNIX-based systems, and in a variety of design methods. Mr. LeJacq has been using object-oriented languages and modeling systems since 1984 for clinical, managed care, client management, engineering and manufacturing design, and aircraft control simulation applications.

JASON MATTHEWS

on transitioning to enterprise component technology

Jason Matthews is cofounder of Genesis Development. He has nearly 20 years of technical and management experience in software development and related professional services. Mr. Matthews is a pioneer in the use of component/object technology for large-scale distributed systems and the Internet, and a specialist in the process of transitioning large organizations to component technology. Mr. Matthews has managed end-user information systems organizations and the development of commercial software products. He has been a consultant to a wide range of industries, including financial services, insurance, healthcare, manufacturing, telecommunications, and energy.

JAMES ODELL

on object-oriented methodologies and agent technology

James Odell was an early innovator of information engineering methodologies, and has spent most of his 30-year career developing better methods to understand, communicate, and

SENIOR CONSULTANTS

manage system requirements. Working with the Object Management Group (OMG) and other major methodologists, Mr. Odell continues to innovate and improve object-oriented methods and techniques. He participated in the development of the UML, and is the cochair of both the OMG's Object Analysis and Design Task Force as well as the Agents Work Group. Formerly, Mr. Odell was the principal consultant for KnowledgeWare, Inc., where he pioneered the concepts of data modeling, information strategy planning, and CASE technology application. Mr. Odell has coauthored several books with James Martin, including the most recent title *Object-Oriented Methods: A Foundation, UML Edition*.

CHRIS PICKERING

on e-business trends and strategies

Chris Pickering, president of the research and consulting firm Systems Development, Inc., analyzes industry practices. Mr. Pickering's areas of focus include information architecture, business-IT alignment, technology acquisition and deployment, organizational change, system modeling, and software practices. He is the author of the survey-based study *E-Business Trends, Strategies, and Technologies* and the periodic *Survey of Advanced Technology*, which tracks the use of advanced information technologies, assesses the effectiveness of that use, and identifies the benefits and hazards of using the leading technologies. He then applies the lessons learned from the research to helping clients maximize their information technology investments. Mr. Pickering's articles and his research findings have appeared in leading industry magazines and books, and he is a speaker at a variety of software conferences.

JOHN R. RYMER

on tools, middleware, and application development for distributed systems

John Rymer is president of Upstream Consulting, which he founded in 1997. Mr. Rymer is a well-known strategy advisor and a veteran industry analyst. Since 1989, Mr. Rymer has developed a strong track record of helping software companies solve difficult market and technical problems. He specializes in application development technology for distributed systems, including tools and middleware. Mr. Rymer is a former vice president and founding analyst at Giga Information Group, Inc., where he was responsible for tracking application development technology and products. Mr. Rymer has been a keynote speaker at *OOPSLA, Networld + Interop, ObjectWorld,* and other industry conferences.

GREG SABATINO

on architecting and implementing highly scalable distributed e-business solutions

Greg Sabatino, cofounder of NetNumina Solutions, specializes in the architecture and implementation of highly scalable distributed e-business solutions. Mr. Sabatino's career has centered on the training, support, and delivery of distributed computing architectures and applications for IT organizations worldwide. His efforts focus on enabling organizations to successfully integrate and employ emerging technologies in order to realize a strategic advantage. Mr. Sabatino's experience spans the retail, petrochemical, telecommunications, pharmaceutical and, especially, finance industries. Mr. Sabatino contributes to several industry publications and speaks at conferences on a variety of distributed computing issues.

KENT SEINFELD

on enterprise information architecture development

Kent Seinfeld is the founder of Enright Consulting, a small group of senior IT consultants. Mr. Seinfeld specializes in enterprise information architecture development. Mr. Seinfeld is a former senior vice president of IT and served in three different positions with CoreStates Bank. He was the founder and manager of the Technology Planning and Research group at CIGNA, a global insurance and financial service company, where he was responsible for computing standards, security policy, development methodologies, and the research and development program. Mr. Seinfeld was the CIO for Girard Bank. Earlier in his tenure he was the principal architect in the design and implementation of a large-scale highly integrated banking system. This system evolved into the foundation of one of the first large ATM networks.

ROGER SESSIONS

on distributed middle-tier technologies

Roger Sessions is the world's leading expert on Microsoft's distributed middle-tier technologies, including COM, DCOM, and MTS. Prior to starting his company, ObjectWatch, Inc., Mr. Sessions worked at IBM, where he was an architect of one of the CORBA services. He was also a lead architect for IBM's implementation of the CORBA Persistence Service, gaining an unparalleled perspective on middle-tier technologies. Mr. Sessions has written four books; his most recent is *COM and DCOM: Microsoft's Vision for Distributed Objects*. He writes the highly respected and often controversial online *ObjectWatch Newsletter*. In addition to frequent speaking engagements worldwide, Mr. Sessions writes articles for many industry publications.

ED YOURDON

on object-oriented design and analysis

Ed Yourdon is widely known as the lead developer of the structured analysis/design methods of the 1970s. He was a codeveloper of the Yourdon/Whitehead method of object-oriented analysis/design and the popular Coad/Yourdon OO methodology. He is also the editor of the *Cutter IT Journal*. Mr. Yourdon is currently focused on issues of business-IT alignment; mitigating risks of large outsourcing initiatives; auditing of large, risky projects; and the development and implementation of e-business initiatives as well as forecasting and tracking critical business-IT "megatrends" in the coming decade.

Mr. Yourdon began his career in the computer industry at Digital Equipment Company more than 35 years ago. Mr. Yourdon is currently a member of the Airlie Council, a group of high-end advisors formulating software "best-practices" for the US Department of Defense. Mr. Yourdon has authored more than 200 technical articles; he has also written 25 computer books since 1967.

CUTTER CONSORTIUM
37 Broadway, Suite 1, Arlington, MA 02474-5552, USA
Tel: +1 781 641 5118; Fax: +1 781 648 8707
consortium@cutter.com
www.cutter.com/consortium/