South African Journal of Information Management
[DoE accredited]
Student Work Vol.4(4) December 2002

Recasting hypertext markup language in extended text markup language: is it the foundation for the future of e-commerce?

M. van der Westhuizen
Technikon Pretoria
Post Graduate Diploma in Information Management
RAU University
mattievdw@absamail.co.za

Contents

1 Introduction

Hypertext markup language (HTML) and extensible markup language (XML) are the lingua franca for publishing on the World-Wide Web.

Having gone through several stages of evolution, today's HTML has a wide range of features reflecting the needs of a very diverse and international community wishing to make information available on the Web.

These questions are asked: Should information professionals have basic HTML or XML capabilities and a understanding of the Web's language? And is HTML or XML the foundation of the future of electronic commerce?

According to the Business Internet Consortium (2002), e-business architects and business managers, who are responsible for strategy and implementing business to business (B2B) solutions, need a basic understanding of HTML or XML as they plan for implementations and develop roadmaps for future business-to-business systems.

The focus of this research was on high-level components of both enabling technologies and business processes for B2B automation. The aim was not, however, an architecture description for direct implementation, and the research did not address the details of the logical and physical models for e-commerce, or implementation details for each layer.

In this article, the primarly purpose is to describe the architectural components needed in a B2B environment. This is a high level view of B2B architecture aimed at delivering a framework for B2B standards convergence and interoperability.

top

2 HTML vs. XML in B2B interoperability

XML is the current hype in electronic business economy. With the down-turn in today's economy, Web services shine on organizations like a beacon of hope. Fueled by Microsoft's Net initiative and IBM's WebSphere platform, Web services promise a new level of compatibility across multiple technology platforms (Bloomberg 2002).

2.1 Web services begin with XML

This straightforward language enables different systems to talk to each other. Web services use XML to create a set of industry-standard protocols for describing and exchanging information and handling transactions between companies. Suddenly system integration is easier and less expensive. Today's Webmasters, Internet technology excutives, information managers, and e-business managers, must understand what Web services are, how they can be leveraged in system integration efforts within, and outside the firewall, and how they can help improve the bottom line.

2.2 HTML is a standard generalized markup language (SGML)

HTML is an SGML application and is widely regarded as the standard publishing language of the World-Wide Web.

2.3 International Organization for Standardization (ISO) HTML

ISO HTML language is an application of the International Standard ISO 8879. ISO, founded in 1947, is a worldwide federation of national standards bodies from some 100 countries. Among the standards it fosters is Open Systems Interconnection (OSI), a universal reference model for communication protocols. Many countries have national standards organizations such as the American National Standards Institute (ANSI) that participate in and contribute to the development of ISO standards making.

According to ISO, 'ISO' is not an abbreviation. It is a word, derived from the Greek isos, meaning 'equal', which is the root for the prefix 'iso' that occurs in a host of terms, such as 'isometric' (of equal measure or dimensions) and 'isonomy' (equality of laws, or of people before the law). The name ISO is used around the world to denote the organization, thus avoiding the assortment of abbreviations that would result from the translation of 'International Organization for Standardization' into the different national languages of members. Whatever the country, the short form of the organization's name is always ISO.

SGML is a language for describing markup languages, particularly those in electronic document exchange, document management and document publishing. HTML is an example of a language defined in SGML. SGML has been around since the middle 1980s and has remained quite stable. Much of this stability stems from the fact that the language is both feature-rich and flexible. This flexibility, however, comes at a price, and the price is a level of complexity that has inhibited its adoption in a diversity of environments, including the World-Wide Web (W3C 2002).

HTML, as originally conceived, was to be a language for the exchange of scientific and other technical documents, suitable for use by non-document specialist. HTML addressed the problem of SGML complexity by specifying a small set of structural and semantic tags suitable for authoring relatively simple documents. In addition to simplifying the document structure, HTML added support for hypertext. Multimedia capabilities were added later. In a remarkably short space of time, HTML became widely popular and rapidly outgrew its original purpose. Since HTML's inception, there has been rapid invention of new elements for use within HTML (as standard) and for adapting HTML to vertical, highly specialized, markets. This plethora of new elements has led to interoperability problems for documents across different platforms.

2.4 XML is the shorthand name for extensible markup language
According to different authors form the W3C (2002), XML was conceived as a means of regaining the power and flexibility of SGML without most of its complexity. Although a restricted form of SGML, XML nonetheless preserves most of SGML's power and richness, and yet still retains all of the SGML's commonly used features. While retaining these beneficial features, XML removes many of the more complex features of SGML that make the authoring and design of suitable software both difficult and costly (W3C 2002).

Many arguments in support of XML have taken strength from a critical appraisal of HTML. HTML's limitations have fuelled the call for and interest in a technology such as XML.

2.5 HTML is a fixed tag set
HTML is a fixed tag set:

- It only describes documents of a single type;
- HTML data are hard to process;
- browsers have permitted all manner of HTML messiness to pass, unchecked, into semi-permanent residency in cyberspace; and
- HTML documents that aspire to function like applications are clogging the Internet with client-to-server traffic.

XML will change all of this, to be sure. However, no one can realistically expect the volumes of active, useful HTML pages to become irrelevant overnight. In fact, HTML has an important role to play in the brave new world of XML. What is that role?

2.6 HTML certainly has its past and current relevance
HTML provides a simple way of structuring hypertext documents and of placing references in one document which point to another. These references, called 'links', may be presented to readers of a document in such a way that a simple 'click' summons the linked document, which is then presented to the reader. The reader has the impression of moving from one document to another. This simple user interface has been wildly successful and as a result the World-Wide Web, the 'Web', has become extremely popular.

2.7 XML data are smart data
In HTML:
```
<p>P266 Laptop
<br>Friendly Computer Shop
<br>$1438
```

In XML:

```
<product>
<model>P266 Laptop</model>
<dealer>Friendly Computer Shop</dealer>
<price>$1438</price>
</product>
```

Both may look the same in your browser, however, HTML tells how the data should look, but XML tells you what it means (Goldfarb 2002).

With XML, your browser knows there is a product, and it knows the model, dealer, and price. From a group of these it can show you the cheapest product or closest dealer without going back to the server.

Unlike HTML, with XML you create your own tags, so they describe exactly what you need to know. Because of that, your client-side applications can access data sources anywhere on the Web, in any format. New 'middle-tier' servers sit between the data sources and the client, translating everything into your own task-specific XML.

Nevertheless, XML data are not just smart data; they are also smart documents. That means when you display the information, the model name can be a different font from the dealer name, and the lowest price can be highlighted in green. Unlike HTML, where text is just text to be rendered in a uniform way, with XML text is smart, so it can control the rendition (Goldfarb, 2002).

Moreover, you do not have to decide whether your information is data or documents; in XML, it is always both at once. You can do data processing or document processing or both at the same time.

In the frenzy of the growth, much of the discipline and good practices of the mature SGML world has been lost, and browser developers have added additional features to the markup language such as new tags and new semantics for tags. As a result, many documents have been created which can only be rendered faithfully on a limited number of browsers. Common Web practice is to hide any syntactic problems detected by the browsers and thus the reader is not aware that a page being browsed is not always faithful to the original authored document.

The International Standard was developed in an effort to ensure that it will remain possible for an author to produce simple hypertext for the Web and be confident that a conforming browser will be able to render the document faithfully. ISO/IEC 15445 represents a core of the language to be supported by all conforming browsers, authoring and validating systems.

top

3 Architectural layers of the Internet economy and the use of XML in B2B interoperatibility

There is a natural structure or hierarchy to the Internet economy that can be directly traced to how businesses generate revenues. Based upon this type of structure, we broadly classify the Internet economy into infrastructure and economic activity categories.

3.1 What is content and how do you manage it?

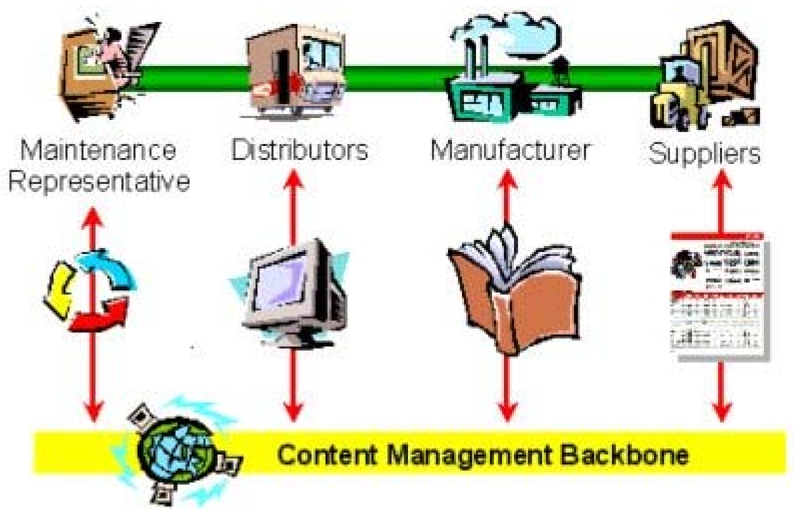
In understanding content management, it may be helpful to distinguish it from document management and knowledge management. Document management deals with maintaining and storing documents. Knowledge management is concerned with making information accessible for decision-making through index, query and search mechanisms. While content management shares some of the attributes of both document management (storing information) and knowledge management (accessing information), it goes beyond them to create a system for re-purposing and using information to drive business processes.

Because 'content' encompasses a wide variety of information objects, an expanded repository and a new system of linking are required. The traditional method of transfering a document into a document management repository will not work with a live feed. Nor will the standard method of linking, object linking and embedding (OLE). In fact, the most common method of reusing content is the familiar cut-and-paste technique and does not involve linking. Content management requires a new enabling technology to accommodate the dynamic array of information (Interleaf 2002).

By standardizing on XML, enterprises will be able to trade with anyone, any time, without the need for the costly custom integration work that has been necessary in the past. But this vision of XML-based 'plug-and-play' commerce is overly simplistic. Of course XML can be used to create electronic catalogues, purchase orders, invoices, shipping notices and other documents needed to conduct business. But XML by itself does not guarantee that these documents can be understood by any business other than the one that create them. XML is only the foundation on which additional standards can be defined to achieve the goal of true interoperability.

Figure 1 (Interleaf 2002) shows how an effective content management system can help a company improve customer satisfaction and loyalty, increase revenue and compete with third-party and after-market parts suppliers.

Figure 1 Content management backbone



Content management encompasses a set of processes and technologies, enabling the creation and packaging of content (documents, complex media, applets, components, etc.) as part of a dynamic and integrated Web-centric environment.

Ultimately, the definition is decidedly Web-oriented, bringing up two questions: Can content be shared between Web and non-Web uses? Are there new technologies that can assist in this process?

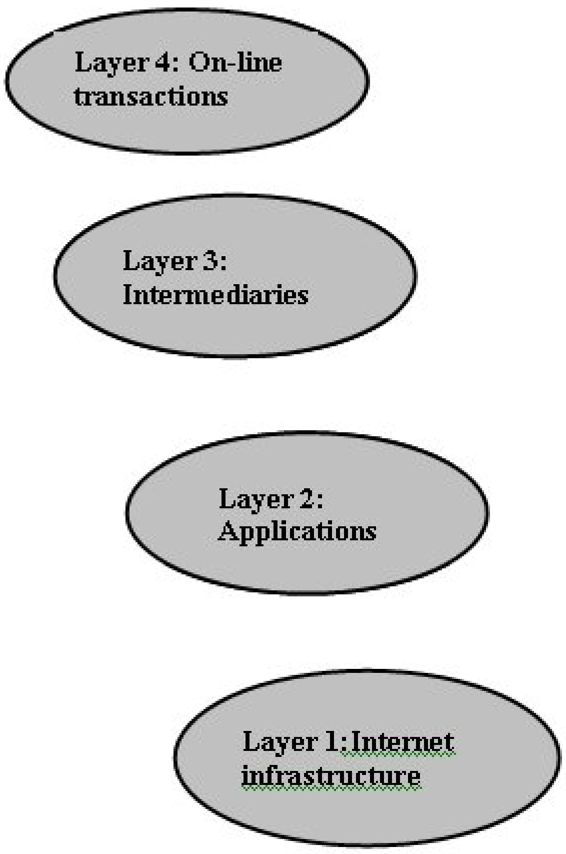
First, content management requires new, enabling technology like XML. Second, content is not merely documents and words, but graphics, audio, video clips, live feeds and software components.

Enterprise content management emphasizes the need to address content management across all forms and formats of information stored throughout the extended enterprise. Within the enterprise, information is created using a wide variety of methods and tools, and this information is revised frequently. Enterprise content management takes the smallest, most appropriate units of information and allows them to be re-purposed and delivered in a personalized form to the individual requiring information.

3.2 Architectural layers in the Internet economy

Figure 2 (Internet Indicators 2001) describes the different architectural layers for publishing content on the Web and XML is often described as the colloquial speech of electronic commerce.

Figure 2 Architectural layers in the Internet economy


Layer 4: On-line transactions
Layer 3: Intermediaries
Layer 2: Applications
Layer 1: Internet infrastructure

The infrastructure category is further divided into two distinct but complementary 'layers'. The Internet infrastructure layer, which provides the physical infrastructure for e-commerce, and Internet applications infrastructure, which includes software applications, consulting, training and integration services that build on top of the network infrastructure and which makes it feasible for organizations to engage in on-line commerce.

The economic activity categories are also subdivided into two layers: electronic intermediaries and on-line transactions.

The intermediary layer involves the role of a third party in a variety of capacities: market maker, provider of expertise or certification that makes it easier for buyers to choose sellers and/or products, search and retrieval services that reduce transaction costs in an electronic market, and other services that facilitate the conduct of on-line commerce.

The transactions layer involves direct transactions between buyers and sellers such as manufacturers and e-tailers. While we could take the position that e-tailers are also an intermediary between the consumers and manufacturers, the illustration above highlights the difference between an e-tailer like Amazon.com and an electronic intermediary in the purest sense of the term. An intermediary would also specify where a book or CD could be found at the lowest price or shortest delivery time or some combination of criteria specified by the consumer. By contrast, Amazon.com only displays its own catalog, prices, availability and lead-time. Of course, it is true that for the case of e-tailers the difference between the two topmost layers could be a matter of degree.

3.2.1 Layer one: the Internet infrastructure indicator

A physical economy critically depends on an efficient infrastructure involving transportation, energy, raw materials and skilled workforce (Internet Indicators 2001). Likewise, the growth of a digital economy depends on the ubiquitous presence of high speed and intelligent electronic networks and the ability to share any type of content between all agents in the economy. The importance of XML can therefore be understood. Accordingly, the Internet Infrastructure Layer includes companies that manufacture or provide products and services that make up the Internet network infrastructure. This layer includes companies that provide telecommunications and fiber backbones, access and end-user networking equipment necessary for the proliferation of Internet-based electronic commerce. This layer includes the following types of companies (Internet Indicators 2001):

- National and regional backbone providers (e.g. Qwest, MCI WorldCom);
- service providers (e.g. America Online (AOL), Earthlink);
- network equipment for backbones and service providers (e.g. Cisco, Lucent, 3Com);
- conduit manufacturers (e.g. Corning); and
- server and client hardware (e.g. Dell, Compaq and HP).

3.2.2 Layer two: the Internet applications infrastructure layer

Products and services in this layer build upon the above IP network infrastructure and make it technologically feasible to perform business activities on-line. In addition to software applications, this layer includes the human capital involved in the deployment of electronic commerce and e-business applications. For example, Web design, Web consulting, and Web integration through XML are considered as a part of this layer. This layer includes the following categories (Internet Indicators 2001):

- Internet consultants (e.g. MarchFIRST, Scient);
- Internet commerce applications (e.g. Microsoft, Sun and IBM);
- multimedia applications (e.g. RealNetworks and Macromedia);
- Web development software (e.g. Adobe, Allaire and Vignette);
- search engine software (e.g. Inktomi and Verity);
- on-line training (e.g. Sylvan Prometric and SmartPlanet);
- Web-enabled databases (e.g. Oracle, IBM DB2, MS SQL Server – only Internet/Intranet related revenues are counted here);
- network operating systems;
- Web hosting and support services; and
- transaction processing companies.

3.2.3 Layer three: the Internet intermediary indicator

Internet intermediaries increase the efficiency of electronic markets by facilitating the meeting and interaction of buyers and sellers over the Internet. They act as catalysts in the process through which investments in the infrastructure and applications layers are transformed into business transactions.

While much has been written about a large-scale disintermediation in the transformation of the physical to the digital economy, the Internet necessitates a new breed of intermediaries whose roles are naturally information and knowledge intensive (Timmers, Stanford-Smith and Kidd 1998).

In the physical world, intermediaries are distributors and dealers, whose primary role is to increase the efficiency of distribution and to lower buyer transaction costs by locating close to the customer population. In sharp contrast, physical proximity is not an issue on the Internet; on-line search, evaluation, communication, coordination, assurance of vendor and product or service quality are the important aspects in the Internet economy. Internet intermediaries play a critical role in filling information and knowledge gaps, which would otherwise impair the functioning of the Internet as a business channel. This layer includes:

- Market makers in vertical industries (e.g. VerticalNet and PCOrder);
- on-line travel agencies (e.g. TravelWeb and Travelocity);
- on-line brokerages (e.g. E*trade, Schwab.com and DLJ direct);
- content aggregators (e.g. Cnet and Cdnet);
- portals or content providers (e.g. Yahoo! and Excite);
- Internet advertisement brokers (e.g. DoubleClick and 24/7 Media);
- on-line advertising (e.g. Yahoo, ESPN Sportszone); and
- Web-based virtual malls (e.g. Lycos shopping).

3.2.4 Layer four: the Internet commerce indicator

Internet commerce includes companies that generate product and service sales to consumers or businesses over the Internet. This indicator includes on-line retailing and other B2B and business-to-consumer (B2C) transactions conducted on the Internet:

- E-tailers selling books, music, apparel, flowers, etc. over the Web (e.g. Amazon.com, and 1-800-flowers.com);
- manufacturers selling products directly such as computer hardware and software (e.g. Cisco, Dell, and IBM);
- transportation service providers selling tickets over the Web (e.g. Delta, United and Southwest);
- online entertainment and professional services (e.g. ESPN Sportszone and guru.com); and
- shipping services (e.g. UPS and FedEx).

It is important to note that many companies operate at multiple layers. For instance, Microsoft and IBM are important players at the Internet infrastructure, applications, and Internet commerce layers, while AOL and Netscape has businesses that fall into all four layers. Similarly, Cisco and Dell are important players at both the infrastructure and commerce layers.

Even though the four-layer Internet economy framework makes it time-consuming to separate revenues for multi-layer players, the framework presents a more realistic and insightful view of the Internet economy than a monolithic conceptualization that does not distinguish between different types of activities. Further, the multi-layered approach lets us analyse how companies choose to enter one Internet layer and later extend their activities to the other layers.

Each layer of the Internet economy is critically dependent on every other layer. For instance, improvements in layer one can help all other layers in different ways. As the IP network infrastructure turns to broadband technologies, applications vendors at layer two can create multimedia applications that can benefit from the availability of high bandwidth. Companies at layers three and four can benefit from improvements in both layers one and two – providing media-rich content to consumers as well as new digital products and service (information and software delivered on-line). This interdependence also exhibits itself in the form of alliances where conduit and content providers or applications vendors and e-tailers join hands to create bundled offerings that are valuable to consumers (Internet Indicators 2001).

top

4 Basic architectural needs to conduct B2B over the Internet using XML standards and Web services today

The task of creating a universal XML business language is a challenging one. Most large enterprises have already invested significant time and money in an e-business infrastructure and are relactant to change the way they conduct electroninc business. Furthermore, every company has different requirements for the information exchanges in a spesific business process, such as procurement and supply-chain optimalisation. Universal business language (UBL) envisions a world where all companies, large and small, can interact seamlessly with their trading partners as if they were part of the same virtual enterprise. It achieves that goal by standarizing the form of information exchange.

UBL envisions a world where all companies, large and small, can interact seamlessly with their trading partners as if they were part of the same virtual enterprise. It would achieve that goal by standardizing the form of information exchange. The lack of a standard for business documents is not due to a shortage of specifications but rather to an overabundance. A multitude of XML business libraries are already in existence. And this has created a big interoperability problem for both users and system vendors. A company that adopts one of these specifications is likely to find that many of the companies with which it would like to trade are inaccessible due to incompatible definitions and XML encoding for many of the same ordinary information elements – product and business descriptions, measurements, dates, locations, and so on. Since use of any e-commerce standard requires significant investment, this 'Tower of Babel' greatly increases both the cost of integration and the cost of commercial software.

UBL proposes a single ubiquitous language for business communication that takes into account both the requirements common to all enterprises and the specific needs of companies in different regions and different vertical industries. Convergence on a single standard will significantly reduce the cost of integration. By lowering the bar to adoption of e-business technology, UBL promises to extend the efficiencies of automated enterprise resource planning (ERP) systems beyond the individual enterprise (UBL 2001).

4.1 Documents, components and context

The primary deliverable of UBL is a set of standard formats for common business documents such as invoices, purchases orders and advance shipment notices. These formats are designed to be sufficient for the needs of many ordinary business transactions and, more importantly, to serve as the starting point for further customization. To enable this customization, the standard document formats will be made up of standard 'business information entities', which are the common building blocks (addresses, prices, and so on) that make up the bulk of most business documents. Basing all UBL document schemas on the same core information entities maximizes the amount of information that can be shared and reused among companies and applications. In a UBL-enabled world, companies publish profiles of their requirements for the business documents involved in specific interactions

(UBL 2001). These profiles specify the business context of each transaction, that is, specific parameters such as the industries and geographic regions of the trading partners. The context parameters are applied to the standard formats to create new formats specific to a given transactional setting. Since these context-specific formats are based on the standard components, interoperability is guaranteed while taking into account the requirements of each party to a particular transaction.

4.2 Opening e-commerce to small businesses

International agreement on a concrete XML syntax for business documents is the key to bringing the majority of the world's businesses into electronic commerce (UBL 2001).

XML markup (the 'tags') transforms documents into hierarchical sets of information objects with logical 'handles' that can easily be manipulated by simple pattern matching and text processing tools like Perl, Python and Emacs. And free lightweight parsers can apply rigorous structural and semantic validation to XML documents to ensure interoperability.

These two characteristics of XML documents mean that when UBL arrives, any reasonably computer literate person with a PC and some free software tools will be able to interact with the UBL-compliant purchasing system of a Fortune 500 company. Custom programming with expensive data extraction and mapping software typical of electronic data interchange (EDI) implementations will still be possible but no longer required. And taming the problem of context-driven data requirements – met in EDI with implementation guides – should greatly reduce the cost of adapting applications to the requirements of particular trading relationships.

International agreement on a single vocabulary for electronic business will create an environment where businesses of every size can automate their processes to exactly the degree that they can afford – from manually programmed text-hacking systems at the low end to off-the-shelf software in the midrange to completely automated and integrated purchasing systems at the high end (UBL 2001). In addition, it will allow all of these businesses to interoperate as if they were technologically at the same level.

The growth of the World-Wide Web demonstrated the inherent power of a standardized tag set running over a ubiquitous transport layer. UBL plus ebXML messaging services will allow businesses of every kind access to electronic commerce, just as HTML plus hypertext transfer protocol (HTTP) allowed publishers of every kind access to the Internet. Like HTML, UBL will someday be seen as a transitional technology. And like HTML, its effect will be revolutionary.

4.3 Example of the impact of XML on everyday lives

Matty arrives at work at eight, grabs a coffee and sits down in front of her computer. Her automated agent has posted a message on her active desktop, with links to a short company profile, the company's 90-day stock history, and its Web site. All of this information has been formatted according to her personal style sheet.

The company, named simply 'E-NTertainers', specializes in on-line entertainment. Matty runs a full-text search on the company's name in her document repository of press releases and news articles and finds three links. The most recent article explains that 'E-NTertainers' is engaged in talks with Steve Hofmeyer about future rights to his interactive music videos. If the deal goes through this would mean dramatically increased revenues and visibility for 'E-Ntertainers'.

Matty presumes that the deal is forthcoming and that word has already started to leak out, pushing up the stock. She drags the category indicated in the original alert, 'on-line entertainment', to her address book, which builds a mailing list of clients interested in companies in this area. She drags the press release and stock history into the message window and adds a short note asking if anyone would be interested in acquiring the stock. A large private investor quickly replies with a request for 20000 shares at the current price of R55,25. She drags the order from this message to her automated ordering agent, which places the order on-line and archives the transaction record, complete with the digital signatures of both buyer and seller.

Three hours later Matty's mailer beeps, triggered by another automated agent instructed to notify her of any new press releases from companies in which she has recently bought stock. Sure enough, the deal between E-NTertainers and Eden has been announced and E-NTertainers stock is shooting up. It reaches R120,25 before levelling off.

Not bad, she thinks, not even noon and I have already made over R1300000 for my client. Just then, she hears another beep. It is her mother's birthday. Matty's calendar program calls up an on-line buying agent, which automatically grabs her mother's mailing address from her address book. It prompts her to enter a short note and then goes out in search of an on-line flower shop with delivery service to Hermanus in the Northern Cape Province. The agent searches for one dozen red long-stem roses, at the best price, including delivery and taxes. As for Matty, she settles back in her chair feeling vaguely guilty about how easy it all is.

Wait a minute! Doesn't this sound a bit too much like science fiction? Just getting two different software applications to work together is close to impossible, let alone a fully integrated array of software agents connected to an active desktop, calendar, mail program, financial information provider, document repository and Internet-based purchasing system. Speaking of which, an automatic purchasing agent has a hard enough time getting consistent price information from different on-line stores; how could it also find out where delivery is available?

While viewing an on-line stock history is no problem, formatting it with a custom style sheet in an e-mail message is very difficult. How could the formatting program figure out how to rearrange the pieces of information according to Matty's style sheet? In addition, wouldn't a full-text query on 'E-NTertainers' retrieve every document in the database?

The technology that is making scenarios like this feasible is XML, which is now clearly emerging as HTML's equivalent for data, according to Hogan (2002). Unlike HTML, XML data elements have well defined tags that describe their content, enabling applications and autonomous agents to extract useful information from them.

In addition, the available tags are not fixed but can be defined in a document type definition (DTD) for a given application. The DTD describes which tags are allowed in the document as well as defining a hierarchy that determines where tags may occur (i.e. a 'paragraph' may occur in a 'chapter', but not vice versa).

Although XML standards are very new, mainstream DTDs for describing software components have already been deployed for applications needing standard-based, open access to information.

Examples of these standard DTDs include Channel Definition Format (CDF) for describing push content and Open Software Description (OSD). Because the investment in XML development tools and expertise can be leveraged across a wide range of applications, communication based on well-defined, standard DTDs will become increasingly important in providing universal access to information across applications from different vendors

(Hogan 2002).

In Matty's case, XML DTDs for address book entries, stock price histories, company information and the like are what enable her diverse set of applications to extract and act on or interact with the relevant information.

Since XML provides a mechanism for tagging data with fields that describe the data, the applications can pinpoint the exact data they need and share it in a manner that is easily brokered between the applications. For example, searching for flowers for Matty's mother simply entails:

- Searching a directory for on-line flower shops;
- connecting to these flower shops and searching for XML tags that correspond to the appropriate delivery area <*Delivery Area*> for Hermanus in the Northern Cape Province;
- Then searching this subset of shops for the appropriate flowers: <*Flower Class*> Rose, <*Colour*> Red, <*Stem*> Long, <*Vase*> none; and
- From this subset, the agent computes the corresponding costs by searching for the information tagged by <*price*>, <*tax rate*> and <*delivery fee*> and doing the appropriate calculation. It then selects and purchases the least expensive offering.

In fact, there is more to this story. Most of the tasks that could be accomplished in this way would occur in the background without any interaction from Matty.

4.4 XML-enabled technologies with specific reference to e-commerce (EC)
The long-expected rise of electronic commerce has been hindered by the difficulty encountered by consumers in finding the desired product among the myriad of vendors setting up shop on the Internet, all with different product lines, prices, on-line viewing capabilities, delivery options and so forth. So-called intelligent agents have not helped because they have an even harder time than humans in trying to make sense of the digital morass presented by HTML.

With XML repository technology, on-line stores can present product information in a standard, structured format, independent-of-page design. Electronic commerce is obviously focused on financial transactions. Using HTML, the user must manually wade through HTML information to extract relevant data like price and tax (Freter 2002).

Unlike text, numbers have no inherent context. In other words, price means something, but how do you know whether a number is associated with a price, a tax, an address, or anything? XML creates this association, making human, and machine interpretation a reality (Freter 2002). XML is the catalyst that will finally unleash the explosive potential of electronic commerce. The XML-aware query facilities of the repository make it possible to retrieve relevant information directly and re-purpose it, as needed for processing by an automatic agent or a user. By reducing the time needed to locate a product, a price or any other relevant information on the Internet, XML repositories will play an important role in making on-line shopping more efficient and enjoyable.

top

5 Universal business language (UBL) – the next step

Two previous efforts in particular lay the foundation of this B2B conceptual model.

Following a layered approach similar to the ISO and Gartner models described in the next sections and taking the latest XML standards and Web services developments into consideration, result in a model that highlights what is needed conduct B2B over the Internet using XML standards and Web services today.

5.1 ISO open EDI view

One of the most widely used attempts at classifying electronic business transaction standards is the ISO open EDI Reference Model (ISO/IEC 14662), as illustrated in Figure 3. The ISO described a two-layer system, defining business transactions in terms of a business operational view (BOV), which focuses on the business aspects of business transactions; and a functional service view (FSV), which encompasses the information technology aspects of these transactions (ISO/IEC 14662 1997).

5.1.1 Business operational view (BOV)

The BOV comprises:

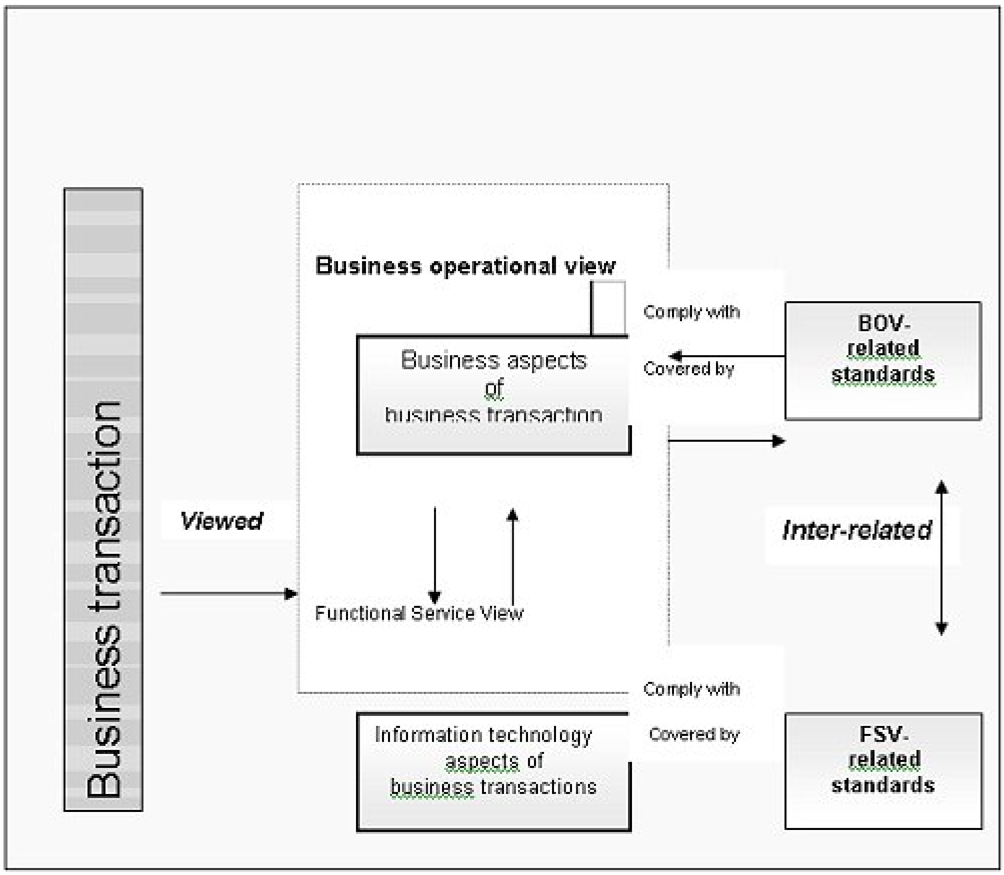
- The semantics of business data in business transactions and associated data interchanges; and
- rules for business transactions, namely operational conventions, agreements and mutual obligations.

5.2.1 Functional service views (FSV)

The FSV comprises:

- Functional capabilities, namely capability of initiating, opening, responding and tracking the progress of transactions;
- service interfaces, such as user application interfaces and transfer infrastructure interface; and
- protocols, namely security mechanism handling; protocols for inter-working of information technology systems of different organizations; and translation mechanism.

Figure 3 ISO open EDI reference model



5.2 Gartner Group view

A view produced by Gartner Group (2000) divides the conceptual architecture into three major sections:

- Top: business content and process standards – meaning of information and processes
- Middle: message and associated structure standards – syntax
- Base: messaging protocol and tools standards – communication.

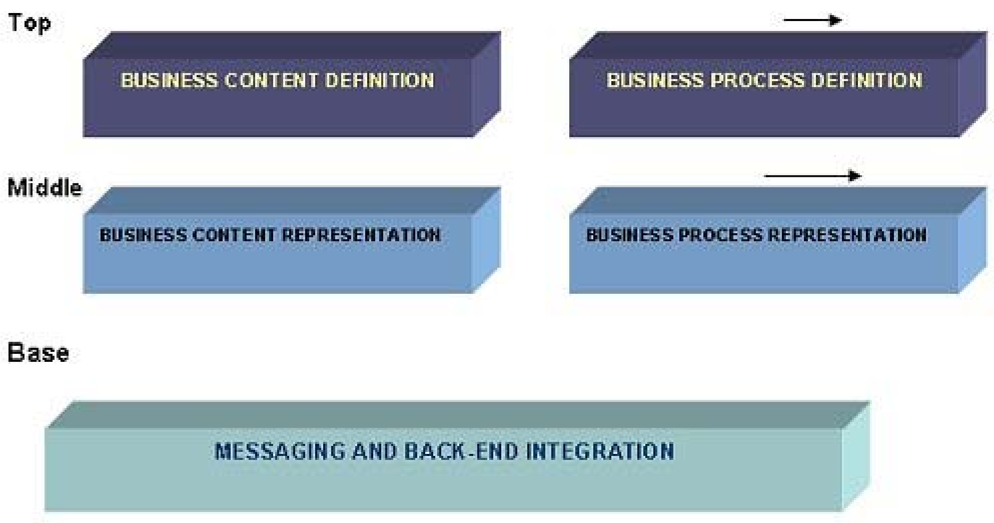
A quote from the Gartner report (Gartner Group 2000) defines the terms as follows:

- *Meaning of information*: the relationship between values in the fields and the external world to which the data relates.
- *Process definitions*: the business rules, the definition of the roles of the parties involved, and the trigger events that provide the context for the exchange of information. Process definitions should cover the complete set of business events required to accomplish a business objective (e.g. placing an order would include steps such as sourcing, issuing a purchase order, receiving acknowledgments and dealing with changes) rather than just discrete steps (e.g. issuing a purchase order).
- *Syntax*: the structure of the message, usually as a sequence of data fields.
- *Communication layer*: the mechanisms by which messages will be transported from party to party.

Based on the categories described above, a high-level functional block diagram of the

B2B architecture appears in Figure 4 (Gartner Group 2000)

Figure 4 Gartner B2B architecture functional blocks


Top
BUSINESS CONTENT DEFINITION
BUSINESS PROCESS DEFINITION
Middle
BUSINESS CONTENT REPRESENTATION
BUSINESS PROCESS REPRESENTATION
Base
MESSAGING AND BACK-END INTEGRATION

Several basic architecture considerations (principles) were employed in developing this model. They are illustrated in Table 1 (Business Internet Consortium 2000)

Table 1 Basic architectural principles

Openness: open standards, open platform.
Layered specifications: enabling technologies provide foundations to higher level functions and business logic; complex problems are divided into manageable layers, applying a divide-and-conquer methodology.
Loosely coupled: supports a loosely coupled approach to integrating trading partners' business applications, as well as B2B other components, connecting using messaging rather than programmable function calls.
Extensibility: architecture can grow over time to cover more business processes and more industries.
Reuse: reuse of business objects and patterns; reuse of technologies and architectural components, not reinventing the wheel.
Self-describing: each component has clearly defined interfaces to describe the services provided and methods for interactions. The description is publishable and searchable on a common registry.
Dynamic discovery and binding: services could be dynamically located on a common registry and composed into more complex services or transactional steps to serve particular business needs.

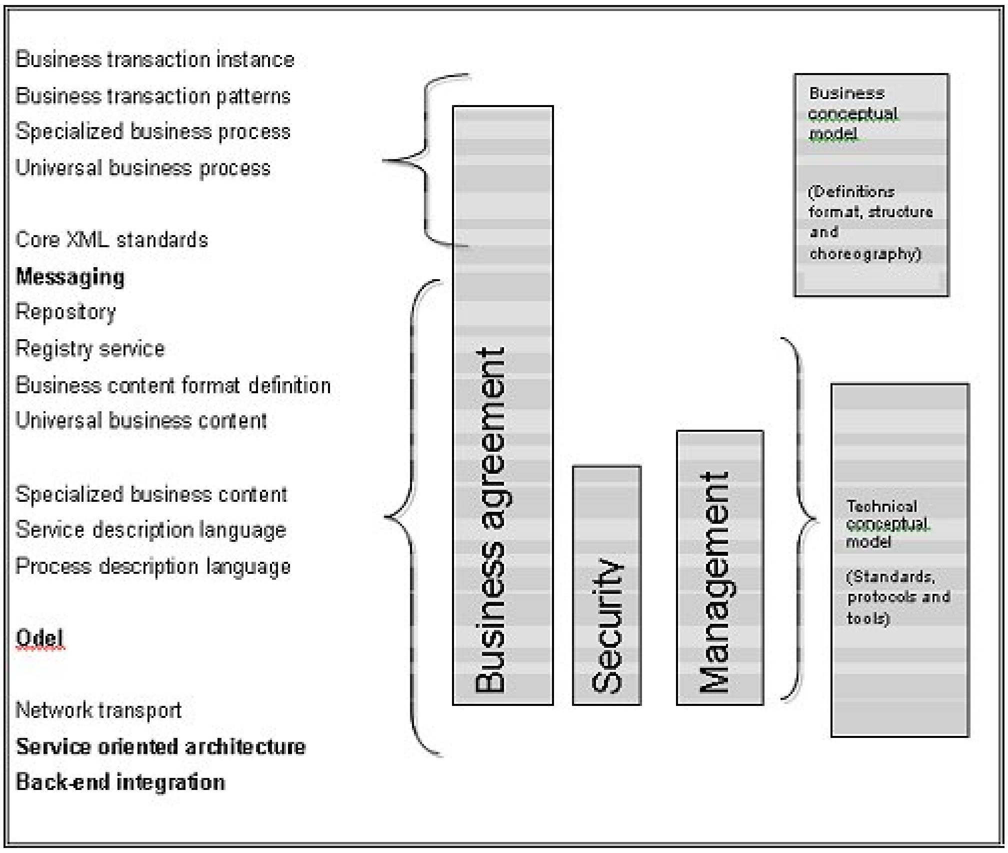
top

6 Conceptual model for B2B integration

Companies across all industries are realizing the fundamental benefits of using the Internet to integrate their supply chains in an automated fashion. The potential to reduce inventory, improve time-to-market, reduce transaction costs, and conduct business with a broader network of supply chain partners has direct and measurable benefits to a company's bottom line.

Figure 5 illustrates the components of the B2B architecture conceptual model for the e-economy (Business Internet Consortium 2002)

Figure 5 Components of the B2B architecture conceptual model for the e-economy


Business transaction instance
Business transaction patterns
Specialized business process
Universal business process
Core XML standards
Messaging
Repository
Registry service
Business content format definition
Universal business content
Specialized business content
Service description language
Process description language
Odel
Network transport
Service oriented architecture
Back-end integration
Business agreement
Security
Management
Business conceptual model
(Definitions format, structure and choreography)
Technical conceptual model
(Standards, protocols and tools)

Because of the benefits that result from supply chain integration, companies are exploring open, XML-based standards to help remove the formidable barriers associated with developing a common business language and protocol for Internet-based collaboration, communication and commerce.

The conceptual model was developed with the input from many industry and technology organizations and respected thought leaders. This model is divided into two general sections: the Business Conceptual Model on the top and the Technical Conceptual Model at the bottom.

The business conceptual model focuses on the definition, structures and formats of business transactions, as well as the business processes that handle transactions.

The technical conceptual model provides the technical foundation that enables business document and process definition. The model is also divided into left and right halves: the left side represents the components that support business content (payload of business transactions), while the right side represents layers associated with business processes.

The three vertically oriented layers (trading partner agreement, security and management) have implications across all, or nearly all, layers.

top

7 Descriptions of the components for architectural e-commerce

Back-end integration provides hooks into the back-end enterprise systems through API or shared messaging bus. Includes functions like business logic processing and format transformation.

7.1 Service orientated architecture – development platform for Web services
This layer provides basic development standards and tools (Java, C#, J2EE, .NET, etc.) and related development environments. This layer also defines APIs that 'glue' e-business transaction systems with the back-end enterprise resource planning (ERP) systems. In this layer we find all the components that help us in writing the code that allows us to connect to our back-end systems but also the business logic of the e-business software itself (Business Internet Consortium 2002).

7.2 Network transport
This layer addresses the basic messaging transport protocols needed to communicate on the Internet, messaging services that provide for asynchronous publish/subscribe, asynchronous message queuing and synchronous request and reply. Additionally, it addresses how messages are placed on and off the transport bus. These standards specify mechanisms for transporting messages in a secure and reliable way (UBL 2001).

7.3 Core XML standards
Basic XML protocols are associated W3C standards for defining document types and for accessing the data within the documents. This syntax is used to express specifications in the layers above for defining the representation of business content and processes (W3C 2002).

7.4 Messaging
These are standardized message and envelope structure and layout definitions, which have specific technical purposes. This layer addresses the need to record session and communication settings for message transport in order to enable coordination between parties in a business transaction, including parameters that control reliable messaging, secured messaging, etc. (UBL 2001).

7.5 Repository
These are standardized repository services that specify the structure and access protocol and schemas for business content storage and retrieval, which includes the term, its constraints, its representations, etc. Many repository initiatives have emerged over recent years. Repositories can provide a very diverse set of functionalities such as the storage of business object definitions, business processes and data-dictionaries (UBL 2001).

7.6 Registry service
This service specifies the structure and access protocol of registries and repositories that trading entities can access to discover each other's capabilities and services. While the repository stores actual objects, the registry provides the management interfaces and protocols to register and discover the entries of a repository. A registry covers naming, directory, search, dynamic binding links, privacy, authentication, authorization, identification services, etc. (W3C 2002).

7.8 Business content format definition
Business content includes everything that composes the payload of business transactions, such as dictionary entries, composition of dictionary entries, special business documents and attachments. Business content format definition is the specification of the data structures, data types, constraints and code lists of all the items necessary to compose valid business content (Kreger 2001).

7.9 Universal business content

This specifies business terminology and accepted values that may be universally used in business messages that support a broad range of industries, business models and locales; the vocabulary used to construct the business content of a message. This content covers many domains of discourse, such as product, materials management, finance and quality (Hall 2001).

7.10 Specialized business content

This refers to industry or supply chain-specific technical lexicon (terms, properties, values, taxonomic structures) to be used to extend and specialize the universal content, to construct the content of an industry-specific business document (Hall 2001).

7.11 Service description language

This layer describes the tools and languages for service implementation and service interface, which is key to achieving loosely coupled architecture and reducing the amount of custom programming as well as the effort of integration between service requester and service provider (Kreger 2001).

7.12 Process description language

This specifies the way in which any business process (whether universal or specific in nature) is recorded, such that it is understood and executable in a repeatable fashion by a wide array of humans and/or applications (Kreger 2001).

7.13 Universal business process

Specifies business processes that are applicable to a broad range of businesses, regardless of the vertical industry or locale within which the business operates or of the specific characteristics of the business. These processes cover many domains of activity that businesses engage in, such as collaborative product development, request for quote, supply chain execution, purchasing and manufacturing (UBL 2001).

7.14 Specialized business process

Specifies business processes are not universally applicable but instead are specific to a business operating within a specific industry or supply chain (such as electronic components, pharmaceuticals or automotive) and locales or business models (UBL 2001).

7.15 Business transaction patterns

These describe allowable (legal) patterns and rules of business content exchanges following certain business processes. The business content used in transactions should be composed of terms defined in the universal business content and specialized business content layers, following predefined patterns and rules. The same is true on the business process side; the process portion of a business transaction is made up of components defined in the universal business process and specialized business process layers, also following predefined patterns and rules (RosettaNet 2001).

7.16 Business transaction instance

This describe an actual instance of a valid or legal business transaction following the patterns defined in the Business Transaction Patterns layer. This is a real conversation and information exchange between two trading partners (RosettaNet 2001).

7.17 Trading partner agreement (TPA)

This determines the dynamic creation and management of relationships between partners, and includes profiles of trading partners' B2B infrastructure, protocols and contractual agreements for transactions (UBL 2001).

7.18 Security

This layer spans a wide range of abstractions from basic encryption, authentication and authorization on the core XML layer, to non-repudiation and security policies in the business process layer. It includes the technologies used to implement both transient and persistent security functions, and the policies that manage and apply these technologies (W3C 2002).

top

8 Examples of some exciting new technologies enabled by XML

8. 1 Internet search engines

Imagine a search engine that understands and uses contextual information when performing a full-text search. Searching for information about the Java programming language would no longer yield links to coffee sites or the Island of Java. This is because searching for the term 'Java' is narrowed down to those fields tagged as a 'programming language'. As a result, the speed and accuracy of the search is dramatically improved. Widespread use of XML repository technology on Web servers will play a vital role in easing the 'information overload' currently suffered by Internet users (Hogan 2002). For example, when searching for information on a subject that is contained in a single chapter or even a single page within a book, XML enables you to retrieve only that chapter or page, while HTML currently gives you the entire book. Of course, all of these benefits require a sophisticated, scalable and fast repository. This repository must be able to manage the rich XML links and understand XML structure so that it indexes text based on its context and use in a document.

8.2 Self-describing binary large objects (BLOBs) and distributed object file systems

File systems today store files as BLOBs, whether they are word processing documents, presentations, databases, pictures, CORBA objects, etc. The information is locked inside the file and is not accessible to the file system. The file system has only minimal information about the files. File systems could do no better with XML data. However, by leveraging a sophisticated object-oriented structure, the contents and structure of XML data can be indexed, searched and manipulated in a sophisticated and granular fashion. As a result, the XML data can be managed in a distributed fashion, much like the Internet itself. A powerful XML linking mechanism also makes it possible to tightly bind a BLOB with a set of XML metadata describing it, sort of a machine-readable summary (Hogan 2002). Since an object database management system (ODBMS) powered XML repository can natively manage structured content, binary data types and arbitrary links between the two, it is the only solution with the power to build distributed object repositories that offer scalable management of legacy documents and XML data.

8.3 Electronic data interchange (EDI)

EDI is done today through secure, value-added networks (VANs) that map the data between companies and their disparate applications. By leveraging XML, the applications easily broker information between themselves. Mapping data from one company's purchasing system to another company's inventory is just a matter of understanding the XML tags on the data (UBL 2001). XML then becomes the universal format for EDI, enabling companies to create ad hoc secure extranets with other companies over the Internet transport, while eliminating the VAN intermediary.

8.4 Data repurposing

By breaking documents into discrete elements, it becomes very easy for individuals to extract the truly relevant information from several sources and reassemble it into any format (e.g. Web page, document or presentation). This helps to address the current information overload, because the user receives only the relevant information. In fact, a personal agent might even assemble the information (Business Internet Consortium 2002). This ability also facilitates the acceleration of learning since it becomes much easier to assemble the 'current' body of work on a particular subject, and then takes it a step further thus pushing the development of human knowledge always forward.

8.5 Content personalization (intelligent pull, agent accumulation and push)
Today, people use the Internet as a news service. By defining a few key words or general topics of interest in specific industries, you can get a fairly good news service. However, this type of service requires human interaction to determine what is actually new. The alternative is to monitor a few news sites. Unfortunately, this approach limits the ability to filter and personalize the information. However, XML, combined with a sophisticated repository, can solve all of these problems. Using XML, you could create a very sophisticated personal news filter that spans multiple sites or the entire Internet. The XML repository would provide the date stamp, enabling agents or search engines to filter the information to extract only the 'new' information. Then, the information could be easily extracted, formatted and delivered in any way that you choose, whether it is an active desktop, a personal news Web page, e-mail or pager (Hogan 2002). This capability will allow all individuals to create 'custom' newspapers with the latest information formatted and delivered any way they want it.

8.6 Customized bandwidth allocation
Allocation and prioritization of bandwidth is an increasingly important issue. The allocation of bandwidth can be accomplished at a number of points in the flow of data, including the information serving company, their access provider, the Internet backbone, the recipient's local service provider, or the recipient's company (of course, this is an oversimplification). At each of these points, bandwidth can be prioritized, but the problem is how to determine the priority and then how to bill accordingly. By attaching the appropriate XML tags, or simply reading the appropriate tags can implement this process in an efficient and consistent fashion (Hogan, 2002). For example, 'urgent' e-mail messages might be tagged urgent and therefore have certain additional rights, also determined by the recipient, such as superior bandwidth, auto-routing to a pager, placing a call to the recipient's cell phone to give a machine translation of the message over the phone, and more. The repository plays a critical role in associating the appropriate routing tags, managing these tags throughout the system, identifying and billing the appropriate customer and more.

8.7 Individual content cache (local cache)
A local XML cache provides a means for more efficient utilization of the facilities enabled by the Internet. As individuals surf the Web, transact business, compile information, communicate, etc., a local content cache could facilitate the process in many ways. For example, as you or your automated agent transacts business, the local cache could provide a dated receipt storage facility. This local cache might also generate an index on the fly as you surf the Internet, enabling you to query your past findings for that gem of a site you now need. A local cache could also store content that is incrementally updated, based on time stamps and element-level delta synchronization. Alternatively, you might choose to cache selected data for off-line browsing (W3C 2002). The exciting opportunities enabled by a local content cache are boundless.

These are just a few of the exciting technologies enabled by XML, and it is easy to understand why XML is creating such excitement in the Internet community. As software developers begin to implement XML applications, however, they will have to address the need to turn these ideas into reality, while keeping up with the ever shortening development cycles characteristic of Web development. In many cases, developers will find that their prototypes work fine in the test laboratory but do not scale to address real world conditions of concurrent usage and data volume. XML's rich interlinking and hierarchical naming structure introduces a whole new set of requirements that bring solutions based on the file system of relational architectures to their knees (Hogan 2002). An XML-wise object repository, designed to be embedded in XML applications of all types, or the operating system itself, is the only solution that provides the functionality and scalability required to drive the realization of this vision of a new generation of networked applications.

top

9 XML repository requirements

All of the exciting applications described above require data persistence, and these requirements are very different from the requirements of monolithic file storage that we are used to. XML extends HTML's simple unidirectional linking, adding support for links to multiple targets, indirect addressing and bi-directionality. Handling this rich linking requires a storage mechanism with far more powerful management of references between objects than that provided by the file system or relational databases. To effectively address the XML opportunity, the storage mechanisms must also be able to understand the structure of XML content, which is composed of a dynamic number of objects, while scaling effectively to handle increased usage load and data volume.

Furthermore, according to the Business Internet Consortium (2002), XML's object-centric focus will create the need for an API that enables rich object-centric manipulation from object-oriented languages like C++ and Java. In other words, what is needed is an XML-aware object repository. Only an ODBMS can maintain information about XML document structure in a scalable manner while handling standard data types and BLOBs with rich hyperlinking and navigation in the database.

9.1 File system storage of XML data

HTML storage management is usually implemented using flat file storage. This is because HTML, lacking any definable structure, is stored as a monolithic block. Using the file system, this way provides acceptable functionality and therefore wins out because it is extremely easy to implement. There are a number of tools that build on the file system's functionality, and file systems are included in operating systems at no additional charge. XML, however, has very different storage requirements. XML applications must store and index the fine-grained elements as well as the document structure. In addition, they must be capable of linking these fine-grained elements directly to each other and to a variety of data types containing associated information (Business Internet Consortium 2002). The increased demands implied by this functionality mean that additional care must be taken to build a system that scales under increasing load. Attempts to parse XML data of any complexity would overwhelm the capabilities of the file system to maintain the rich linking structure and semantics of the data.

Of course, the alternative is to store the XML data as BLOBs and then parse them on the fly each time they are used. This results in sub-optimal performance, because of repeated parsing. In addition, once the data had been parsed, the file system could not execute the complex data manipulation required. In addition, this BLOB storage approach undermines the ability to link various disparate elements into a rich tapestry of information that models real-world usage cases. Of course, implementing all of these features on top of the file system is a possibility, through custom development, but this would essentially recreate object database functionality from scratch.

9.2 Relational database storage of XML data

Relational database management systems (RDBMS) are the other plausible candidate. Unfortunately, their table-based data model is very poorly suited to the hierarchical, interconnected nature of structured XML content. Never the best systems for managing variable length data and BLOBs, RDBMSs are further hampered by the fact that they must represent the tree structure of XML content with an inefficient set of tables and joins. Relational databases disassemble the XML objects in order to fit them into their tabular architecture. As a result the XML object's structure and semantics are either lost, minimizing its value, or must be duplicated in the design of the database.

Duplicating the structure and semantics of complex XML objects in the design of the database is very difficult, particularly if the structure of the XML data is variable, as it usually is. The rigidity of the relational design is a poor fit with the dynamic assembly and manipulation of XML data (W3C 2002). Relational databases also cannot handle object-level locking; the best they can provide is row-level locking. Since relational databases decompose XML elements into various tables, linked via keys, it is very difficult to implement an effective locking scheme that does not dramatically hinder concurrent use and scalability.

In concurrent editing environments, there will be an increase in demand for related objects from disparate users. Relational databases respond by locking entire rows across multiple tables. This can cause unacceptable performance degradation if multiple users are requesting different objects that are locked via this broad locking scheme. If the DBA responds by separating the information into a larger number of more granular tables, the performance is degraded by the number of joins required to model the richly linked structure of structured XML content. In addition, relational databases are typically too weighty to form an infrastructure for embeddable storage and require substantial development to adapt to the complex structure of structured XML content. Quite simply, relational databases, while excellent for many purposes, are not architecturally compatible with the storage needs of XML data.

9.3 Object database storage of XML data

The architecture of object databases is ideally suited to handling XML data; in fact, the adoption of XML data could be the 'killer application' for the object database market. Object databases are designed to handle objects in their native forms. The objects then maintain their own data, methods, relationships and the semantics of the whole model. This is ideal for the creation and management of hierarchical XML trees, while providing both hierarchical tree navigation and rich link traversal (Business Internet Consortium 2002).

For example, traversing to the other side of a tree in the two dimensional relational model forces the developer to climb up the tree, through joins, and then back down the other side. The rich relationship linking of an object-oriented model enables both hierarchical navigation and rapid branch traversal, reducing computation and increasing performance. Object databases are also designed to handle arbitrary, variable-length data types and interrelated data. This is critical due to the various data types linked within structured XML content.

XML also enables an ever-changing Web of relationships between hyperlinked data elements, such as on-the-fly creation of documents. Object databases, with their flexibility and rich relationship management are ideal for managing this type of information.

Those object databases that allow object-level locking, provide for a much more granular locking than relational or file system-based solutions. This granular locking is critical for user scalability, since it limits the conflicts between user requests for data.

Object databases are also designed to handle larger-than-memory content, providing for content scalability. Object databases also offer more simplified creation and management of distributed partitions, which further addresses the issue of database volume scalability and distributed implementation. In short, object databases were designed to address the very requirements that XML is just now starting to force upon tomorrow's storage solutions.

Just as most software developers would never consider developing their own encryption technology for a new product, licensing it instead from a third-party vendor selling best-of-breed solutions, it makes sense for XML developers to seek a third-party storage engine that instantly provides them with the feature set they require for effective XML data management. These applications need a mature database engine well suited to the structure and data types associated with XML.

9.4 Ideal repository for XML data

The ideal XML repository should address the needs of XML as well as the needs of the associated applications. In evaluating the requirements of applications in this field, the following criteria are critical:

- Scalability
- Language support
- Ease of programming
- Embeddability.

Scalability will be very important since the XML applications described above will run on both the client and the server. It is important that the object database scale down as well as up, while leveraging the same APIs, to simplify application development. Language support is also important. Ease of programming is critical due to the compressed development cycles, particularly in the Internet. Embeddability encompasses two criteria – zero-management and low memory footprint. Embeddability is an interesting issue since it has both short and long-term ramifications (Business Internet Consortium 2002).

In the short-term, embeddability is important because most initial XML applications, lacking sufficient XML support in the file system, will build-in this support. However, in the long-term, the object database will replace the standard file manager running on top of the file system. This of course will make embeddability an absolute requirement.

Storing and retrieving XML data are only the start. In addition to standard XML data storage, the ideal repository would offer tightly integrated XML-specific tree navigation, versioning, management of arbitrary links, import and export, publishing of structured content on the Web, support for object-oriented programming languages as well as common scripting languages, and more. Building these facilities on top of a object database are non-trivial, yet they are critical to the actual process of managing and manipulating the XML data stored in the object database.

top

10 Conclusion

E-commerce, enterprise resource planning (ERP), database, file sharing and similar applications are mission critical. When the infrastructure breaks down, these applications stop working, and the losses, both in revenue and productivity, start accruing (Morrissey 2001).

While XML has evolved from SGML and HTML, its impact will not be evolutionary – it will be revolutionary! XML will transform the Internet from a massive collection of unmanageable data into the intelligent transport we have all been waiting for. The development community is just beginning to recognize the far-reaching benefits of XML as a standard, flexible, structured content format. Existing XML DTDs, such as CDF for push channel management and OSD for on-line software distribution, merely hint at the types of applications that will benefit from XML's unique characteristics. The announcement by Bill Gates that future versions of Microsoft Office will support XML virtually assures its broad adoption (Business Internet Consortium 2002).

Because XML applications have a far wider scope than their HTML counterparts, in terms of both application domain and type of data being managed, existing solutions for HTML storage are not sufficient for XML. File system-based solutions do not support XML's rich linking and hierarchical tree structure, negating XML's added value. Relational databases are based on a two-dimensional, table-based architecture that is also ill suited to the needs of XML. Attempts to force a relational database into storing XML will result in sub-optimal performance, concurrent access and scalability due to the architectural mismatch between XML content and relational databases.

The only database architecture that is suited to the demands of XML is the object database. Only object databases support granular element access and locking for superior concurrent access, rich high-performance hyperlinking and fast hierarchical navigation.

top

11 References

Bloomberg, J. 2002. *XML and Web services unleashed*. Wakefield, Maccachusetts: SAMS Publishing.

Business Internet Consortium. 2002. *High-level conceptual model for B2B integration –* XML convergence workgroup. Version 2.0. 12 March 2002.

Freter, T. 2002. XML: document and information management. Sun microsystems. [Online]. Available WWW: http://www.sun.com/980908/XML/.

Gartner Group. 2000. OAGI: fostering standards evolution or revolution? A Gartner advisory research note. 14 December 2000.

Goldfarb, C.F. 2002. XML in an instant: a non-geeky introduction. [Online]. Available WWW: http://www.XML.org/XML/goldfarb.shtml.

Hall, R.E. 2001. W3C Web service position paper from Intel. W3C Workshop, April 12.

Hogan, M. 2002. XML: The foundation for the future. [Online]. Available WWW: http://www.XML.org/XML/XML_foundation_future.shtml.

Interleaf, 2002. Putting XML to work: advantages of content management. [Online]. Available WWW: http://www.xml.org/xml/putting_xml_to_work.shtml.

Internet Indicators. 2001. January 2001 Internet economy indicators. [Online]. Available WWW: http://www.Internetindicators.com/.

ISO/IEC 14662,1997. *Information technology – open EDI reference model*. First edition. December 15.

Kreger, H. 2001. Web service conceptual architecture 1.0. June.

Morrissey, P. 2001. Infrastructure – The survivors guide to 2002. *Network Computing* (December). [Online]. Available WWW: http://www.networkcomputing.com/1226/1226f8.html.

RosettaNet. 2001. RosettaNet architecture conceptual model. July

Timmers, P., Stanford-Smith, B. and Kidd, P. 1998. *Opening up new opportunities for business*. Proceedings of the G8 Global Marketplace for SMEs Electronic Commerce Conference. Manchester, September. Cheshire Henbury, Macclesfield, UK.

UBL. 2001. *UBL: the next step for global e-commerce*. UBL Marketing Subcommittee. 26 December.

W3C. 2002. XHTMLTM 1.0 The extensible hypertext markup language (Second Edition). [Online]. Available WWW: http://www.w3.org/TR/xhtml1/.

ISSN 1560-683X

Published by InterWord Communications for the Centre for Research in Web-based Applications, Rand Afrikaans University