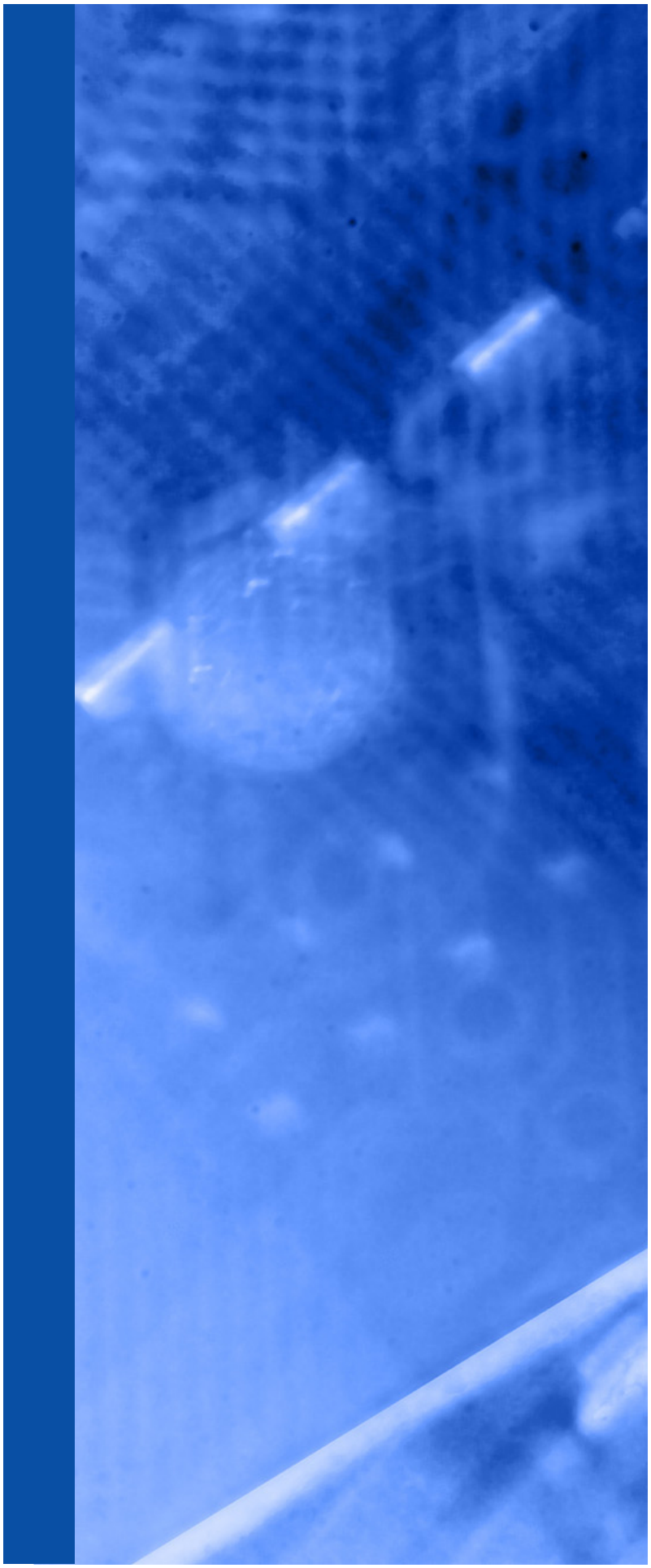


**White Paper**

# Using XML and Databases

## W3C Standards in Practice

February 2008

Bill Trippe, Senior Analyst

Dale Waldt, Contributing Analyst

*Sponsored by*

EMC² where information lives®

THE GILBANE GROUP

**Gilbane Group Inc.**

763 Massachusetts Avenue
Cambridge, MA 02139 USA

Tel: 617.497.9443
Fax: 617.497.5256

info@gilbane.com
http://gilbane.com

# Table of Contents

# Executive Summary

XML has emerged as a powerful format for representing data in a wide variety of fields, from technical data to finance to healthcare. Unlike traditional data formats, such as relational data, XML has a hierarchical structure that can be used to model virtually any type of data. In addition, XML is far more flexible and forgiving of change than other formats.

XML presents a number of interesting challenges and opportunities for data storage. Relational databases and full-text search mechanisms that have been the backbone of many applications are not designed to manage XML content effectively. A new class of databases has emerged that is designed specifically to manage XML content. Typically called "XML Native Databases" or just "XML databases," they incorporate functionality that greatly improves the management, searching, and manipulation of XML to produce the most effective XML data management solution.

The World Wide Web Consortium (W3C), the standards organization that developed XML, has also developed many standards that can be used to access, search, process, and store XML data. XML databases take advantage of these standards to provide efficient and precise access, query, storage, and processing capabilities not found in traditional database technology. The result is that applications using XML databases are more efficient and better suited for managing XML data.

These W3C standards, including XML Schemas, XSLT, DOM, XLink, and XQuery, are well established and tested in real world applications. The XML databases that take advantage of them provide the platform for industrial strength applications to manage XML content.

Like any new technology, adoption is slow at first. Then as the technology matures and understanding on how to best deploy increases, applications emerge that demonstrate the advantages of the approach. Today, we can find many applications to manage XML content that demonstrate the power and flexibility that can only be achieved through XML-native databases. Information intensive companies such as the airline and manufacturer described in this paper have achieved significant technical and business benefits from their use of XML standards and database technology over alternative approaches.

# Introduction

Organizations that manage complex or diverse structured content have discovered the power of XML for expressing the structure of their information. Related standards and tools allow their XML-encoded data to be processed and reused efficiently to satisfy a range of business objectives. Developers are becoming increasingly fluent in using XML, standards related to XML, and a range of platforms and tools to develop XML-based applications. These platforms include relational database management systems that have continued to add more capability for managing XML data and content. Even with this continued innovation and improvement, developers are finding that certain content and data processing requirements are best met with a native XML database technology.

> ***With the right platform and tools, developers can take full advantage of the capabilities of XML and related standards.***

The key to effective XML content management is a flexible system architecture based on interoperability standards for data, services, and system integration. These standards serve as an intermediate middle layer allowing powerful tools to manage, process, access, and transform XML structured content in a cost effective manner.

This paper describes the challenges faced in managing information in robust systems that enable reuse of XML-encoded data objects and support their efficient processing. This is accomplished through the use of XML databases and open standards based on the eXtensible Markup Language (XML) produced by the World Wide Web Consortium.

This paper is divided into three major sections, exploring:

- The Power of Standards-Based Computing,
- Status of Standards used by XML Databases, and
- Two Customer Case Examples

# The Power of Standards-based Computing

In the last twenty years, there has been an explosion in the number and complexity of computing technologies that are available to engineers when creating systems, solutions, and services for managing database content. Many of these technologies, such as niche programming languages, are too specific or limited to be of particular use in a heterogeneous computing environment. Other tools cannot efficiently process structured content such as XML documents. Also, some XML-aware technologies are still in the process of being standardized and are not yet fully reaching their potential or may change once completed, adding risk to the adopter. Even so, many standards exist and are in productive use, enabling computers to support the creation, management, manipulation, and distribution of robustly formatted XML information.

Ideally standards make it easier and quicker to create systems, solutions, and services, since they should enable better, faster integration as well as making it easier to find engineers with requisite knowledge. The plethora of available standards makes this difficult, however. This section of this paper describes some of the problems engineers face when trying to use the most suitable standards and technologies.

## Standards Development Organizations

It is often said that one of the reasons there are so many standards is because there are so many standards organizations, or groups calling themselves such. Having plenty of standards to choose from, in general, is a good thing. Standards enable interoperability and should make it quicker and easier to build quality systems, solutions, and services. There is one big problem with standards, however, and that is that picking the right standard to use for the current problem at the current point in time isn't always easy or clear.

Vendors of computer software often play important contributing roles to the development of standards. Reasons why they participate in the creation of standards include serving their customer base and community at large, promoting the use of a technology they can provide tools for, integrating with their partner's technology, and even sometimes gaining an edge over their competition.

Since there are many reasons for creating standards, it follows that there are also different standards organizations with different missions, standards, and development processes. One of the leading and most recognized of these is the World Wide Web Consortium (W3C). The W3C's vision is "leading the Web to its full potential." W3C standards activity is guided by a vision of the environment and specific standards that need to be developed to lead the Web to this potential.

Others (such as OASIS, the Object Management Group or OMG, etc.) work on other aspects of the overall content processing model, while still others (such as the Joint Photographic Experts Group) work on one standard only serving a specific purpose (in this case a format for graphic images). With the many groups simultaneously working on standards for the different pieces of the processing puzzle, it is no wonder people find it difficult to keep track of the many specifications, their roles, and current stage of development.

There are a number of XML-based standards that are currently available that are used in XML data management. These include familiar languages such as XPath, XML Schema, and the DOM. While these standards are currently adopted by the W3C, there are other areas of functionality not covered by an existing interoperability standard. Even so, the standards that exist today create a powerful architecture for connecting XML data management services to commonly available application frameworks. In short, these XML standards enable the development of powerful XML content management applications to meet even the most demanding business requirements.

## Comparing Relational and XML Databases

Although documents contain useful information, they haven't traditionally been used as data sources. The advent of XML changed this, as each part of a document could now be labeled with exactly the kind of data it contained, enabling targeted searches and the development of more powerful document-centric applications.

Early attempts to manage XML documents often cobbled together full-text search engines, relational databases, and flat files. These early systems suffered from two main problems. The first was scalability, as these systems tended to degrade past a few thousand documents, while many applications involved hundreds of thousands or even millions of documents. The second problem was the lack of structured queries, since the full-text search engines were sometimes not XML aware and queries over metadata were limited to a few pre-selected fields.

Other problems included synchronization between the database and non-database components, the need to write custom code to process results, lack of node-based updates (a problem for large documents), and brittleness in the face of evolving schemas. For those systems that attempted this work without a database platform, they often suffered from the usual laundry list of reasons for why a database should have been used in the first place: concurrency, security, transactional safety, and so on.

Some developers looked to Open Source databases for managing their XML assets. While a few of these have limited XML support, in the form of utilities for exporting relational data into XML formats, they typically do not support native XML data or XML-aware queries. The result is that users of these databases had to build a lot of custom functionality that offset any of the advantages they perceived Open Source databases to have.

Over time, the major relational database vendors worked to address some of the gaps in XML feature coverage, giving developers more tools and functions for modeling the XML data, writing applications, and running queries. The result has been a steady growth in the use of relational databases for XML applications. Still, some of the most challenging applications push the limits of the relational databases and the mechanisms by which they support XML. This is especially true with very large documents, very large collections of documents, and applications where complex document types need complex parsing, manipulation, and querying.

A solution to some of these problems was the introduction of XML databases. When these appeared shortly after XML 1.0 was released, people weren't sure if they were a replacement for relational databases or a return to hierarchical databases. In fact, they were designed for the entirely new types of applications XML made possible.

XML databases have a number of features that are useful for working with XML. The most important are the XML data model, which is flexible enough to model subjects as diverse as technical documentation, health data, and customer profiles; XML-aware full-text searches; and structured query languages like XQuery. They are also designed to manage large numbers and a diverse array of XML documents. Other advantages include node-level updates (which reduce the cost of updating large documents), links, and versioning.

Another advantage of XML databases is their ability to handle large documents, as well as large numbers of documents. Both of these are difficult to query due to the time it takes to parse the documents and find the required data. XML databases solve this problem by parsing and indexing documents when they are inserted. This allows documents to be queried without further parsing and may even allow queries to be resolved only by searching indexes.

A final advantage is more flexibility in handling schema evolution than is found in relational databases. While schema evolution, or changes in the data model, is a normal thing, it can move slowly in the relational world if the particular relational database lacks tools or functions to make changes to the relational schema easier and more manageable. In the XML world, change moves more quickly, both because XML is new and because XML exposes users to more sources of change. Examples of the latter include external trading partners who control the schemas used to move data across organizational boundaries, rapidly evolving fields like finance and biology, and long-lived fields like mortgage and insurance contracts. Fortunately, some XML databases do not require fixed schemas and can easily handle data conforming to multiple schemas or multiple versions of a schema.

Ironically, the strongest endorsement of XML databases to date is that the major relational databases are adding native XML storage capabilities. This shows that the need for and application of native XML data management has become well understood, and adoption increases continually.

## XML Standards and XML Databases

Numerous databases utilize XML processing standards. A reliable and extensible XML database uses both XML standards and XML-aware proprietary technology to deliver high-performance processing regardless of the number of concurrent users, the number of documents or the database size.

An example of a powerful XML database is EMC Documentum XML Store. It is designed for software developers who require advanced XML data processing and storage functionality within their applications. The comprehensive EMC Documentum XML Store Java API contains methods for storing, querying, retrieving, transforming and publishing XML data. EMC Documentum XML Store also has a built-in transaction mechanism that updates both the data and the indexes and that supports load balancing and replication of databases over multiple machines.

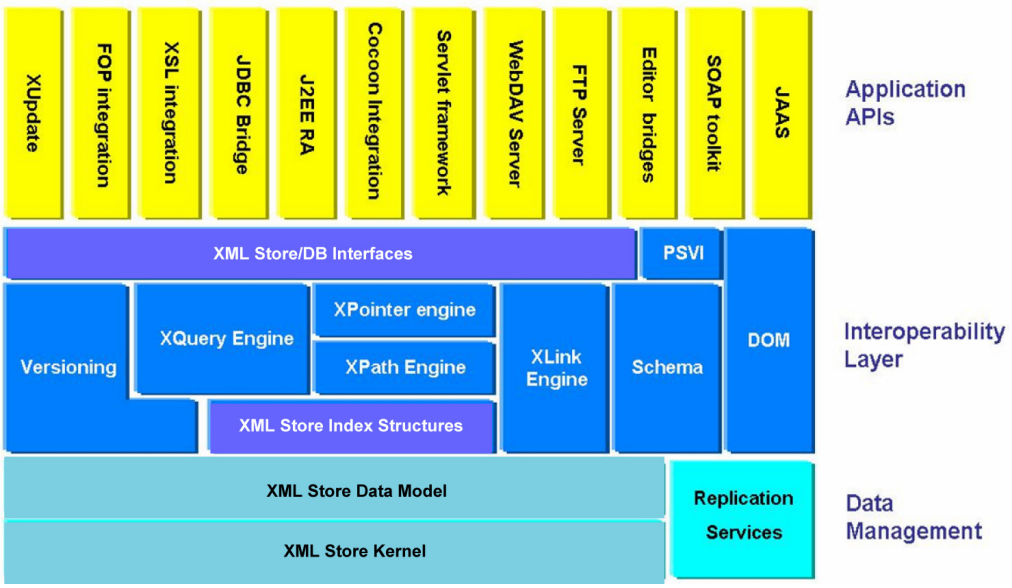
EMC Documentum XML Store provides numerous XML-aware features to enable powerful management of XML data structures, including:

- an XQuery engine for retrieving specific parts of a document
- a versioning mechanism for tracking differences within XML data
- various indexing methods to optimize access to frequently used XML data
- a transformer and formatter for publishing XML data in XHTML or PDF

EMC Documentum XML Store uses and supports XML standards including XML 1.0 and 1.1, XQuery 1.0, XML Schema 1.0, XPath 1.0, XSLT 1.0, XPointer, XLink 1.0, and DOM Level 1, 2, and 3.

The following diagram shows standards and industry specifications that EMC Documentum XML Store works in concert with to create a robust XML data management environment. This diagram is organized into three layers:

- **Application APIs**: Interfaces to commonly available application frameworks used to develop much of today's business and content applications
- **Interoperability Standards**: Open standards that serve as a consistent means for interoperability between specific applications, and
- **Data Management Services**: Specific services to manage XML data content.




But even in this diagram not all aspects of interoperability are currently addressed by existing standards. Work continues in several venues on specifications, either to enhance existing functionality or to create new standards that play a role where no standard currently exists. Standards development for XML data management will continue for the foreseeable future, but there is already a significant amount of the infrastructure illustrated in this diagram in place today with existing standards. A brief update of the existing, emerging, and missing pieces of this standards puzzle follows.

# Status of Standards Implemented by XML Databases

## Roles and Readiness of Standards

A number of standards related to XML databases are being produced under the aegis of the W3C. The roles listed below are based on the mission statements of the standards specifications themselves. The description and categorization of the standard's current status is based on several factors and is not the official word of the W3C or any of its committees. Rather, status is based on the following formula in order to provide some consistency in estimating their readiness, completion, and the risk associated with adoption of the specification.

| Status | Description | Risk |
|---|---|---|
| **Emerging** | Specification is still a working draft in development in a standards committee. It may change substantially before it is passed, or may not achieve the status of a completed specification. | **High** risk associated with adoption of the specification until passage due to potential for change. Should only be implemented when no alternative exists or if it is easy to manage future updates. |
| **Adopted** | Specification has passed as first or subsequent version after completing all requirements of the standards body needed for passage as a standard. Changes will only occur according to the standards development process guiding the development of standards for this organization. The current version is likely to remain stable and in use for some time. | **Medium** risk associate with implementation of this specification since it is untested and subsequent releases may be in development. Most likely several applications have demonstrated the specification's effectiveness and may even have pointed out issues that will be submitted to the originating committee for future consideration and development. |
| **Mature** | Specification has passed as either a first or subsequent version after completing all requirements of the standards body needed for passage as standard. Many implementations have utilized this specification demonstrating its usefulness and stability. | **Low** risk associated with implementation of this specification due to the extensive use and testing it has gone through in implementation. Developers have ample resources to understand its application and any limitations it may have. |

***Table: Standards Status Criteria***

Admittedly the difference between a standard given the "Adopted" and "Mature" status described above may be subjective and dependent on the specific usage and environment. The distinction between the two is to help those not familiar with a particular specification to understand whether there are issues that have yet to be resolved with a completed specification or if there are many tried and true applications of it that demonstrate its readiness. For the most part, the standards described below according to this simple rating system will be mature, well proven standards.

## XML

**Status:** Mature

The eXtensible Markup Language (XML) was adopted in 1998 by the W3C and has enjoyed an explosive growth ever since. Not only have many organizations built applications that maintain content in an XML format, dozens of newer standards have been developed based on XML, or even using XML documents as a description language. It is hard to imagine a language that is more tested and proven than XML.

XML documents can be managed with or without a related set of validation rules. Validation is the addition of a set of rules for element and attribute names, occurrence, and sequence that allow organizations to manage documents with much more custom-defined control. These rules are expressed in either a Document Type Definition (DTD) or an XML Schema, both of which are described in more detail below.

In environments where information must be consistent with structural, naming, and occurrence rules, one of the following schema languages available to validate XML data should be employed to define and enforce these rules.

## Document Type Definitions

**Status:** Mature

Originally XML provided only one mechanism for defining custom data model definitions, which used a specialized syntax defined in the XML specification. The resulting definitions called Markup Declarations are assembled with other modeling information in a Document Type Definition (DTD) and are very commonly used in content applications and other document processing environments. The XML specification includes a syntax for describing custom languages that in turn describe application-specific data structures. An organization, or group of organizations, defines these custom application data structures to serve as a vocabulary and grammar for classes of information that they depend upon to conduct their business.

The data managed according to the rules expressed in DTD are called Document Instances. This is similar to how individual objects are members of a class of objects in Object Oriented design and programming.

A wide range of XML processors and editors running in nearly every known operating environment can reliably and consistently verify that XML data conforms to the rules specified in a DTD or schema. This validation is a key benefit of encoding information in XML as opposed to maintaining and sharing information in other formats.

Organizations use the DTD syntax to define a vocabulary and document structure that supports a specific processing application. Some DTDs model books and the rules for the content they contain. Others model business documents such as invoices or parts catalogs. Any information critical to conducting business can be modeled against the rules for how the information should be created, maintained, and transformed for delivery to other users. The rules expressed in a DTD ensure that all required elements are present before the data progresses to the next step in the process. For instance, it is useful to insist that an invoice must contain a purchase order number before it is sent to the accounts payable department.

## W3C XML Schema

**Status:** Mature

The W3C XML Schema Definition Language, commonly referred to as W3C XML Schema, or even just XML Schema, is one of several languages that provide additional definition constructs for data modeling. The resulting data model is called a schema, and it can define the same element and attribute structures a DTD can define and a great deal more. Additional capabilities include more specific hierarchical rules than a DTD, several data modeling techniques for easing definition and maintenance of a schema, as well as a powerful set of constructs for defining data typing rules for element and attribute content.

These advanced capabilities make XML schemas highly suitable for business system and database applications that were inordinately constrained by the limitations of DTDs. It is useful to mention that there are several alternative XML schema languages in development, but the W3C XML Schema Definition Language is by far the most widely used and adopted at this time.

When the data used in a business system needs to be maintained in a form that is more complex than can be expressed in DTD rules, XML Schema can be used to enforce these rules and ensure that the data meets business processing needs. For example, an organization that creates, stores and manages product specification information (spec sheets) would want to make sure that certain fields of information are maintained in a specific format. This enables the data to move seamlessly between applications and databases.

Examples of the types of rules that can be defined and enforced in XML Schema include the weight or dimensions of a product that need to be in a numeric format with no more than 2 positions to the right of a decimal point. Another example is a rule that ensures that the value entered into an email address follows the style

characteristics of a valid email address. Also, date elements can be tested to ensure that their values are in fact valid dates. These data typing rules can be customized and extended for even the most complex rules an organization wishes to enforce. This level of quality control enforced by XML Schema prevents data errors and avoids excessive data clean up, and the XML Schema definitions used to define these rules are easily shared with and understood by other XML Schema aware applications.

## Namespaces in XML

**Status:** Mature

It is not uncommon for XML data encoded according to several different specific DTDs or XML Schemas to be processed in the same environment. This mixing of different XML vocabularies could cause confusion to processors and human users alike, especially when similar elements from two document types have the same names but different content rules. Namespaces in XML is a specification that allows systems to differentiate between vocabularies and sort out this confusion.

For example, the XLink standard defines structures that can be used in any document to define links to other documents. These structures have their own namespace, so they cannot be confused with similarly named structures in other schemas. This kind of reuse, which is made possible by Namespaces in XML, allows you to take advantage of the effort of other parties.

XML Namespaces is an essential tool in an XML processing environment that removes constraints that would otherwise require expensive and time consuming reformatting of data shared between systems. For instance, a large manufacturer that has processing environments in many locations and configurations would need to share operating and product information between these systems. Namespaces would be used to differentiate the source, and therefore which DTD or Schema is applied, for each data type encountered during processing. The alternative, to reformat all data into a complicated single vocabulary, is usually cost prohibitive and unfeasible.

## XPath 1.0 and XPath 2.0

**Status:** XPath 1.0: Mature, XPath 2.0: Adopted

XPath 1.0 is a language for addressing parts of an XML document and designed to be used by XSLT and other XML processing systems. XPath 1.0 contains basic functions and is in wide use today. XPath 1.0 was adopted as a W3C Recommendation on 16 November 1999.

XPath 1.0 is a language that allows applications to locate and manipulate very specific portions of a document. This precision allows an organization to automate many types of processing and avoid manual document layout and information reformatting.

XPath 1.0 defines several different types of information structures called nodes in an XML document, including elements, attributes, text and comments, as well as powerful node relationships such as ancestors, children, siblings and prior and following occurrence. The result is a powerful expression language used to accurately identify nodes in an XML document to support processing such as XSLT style rendering and data transformation.

XPath 1.0 is an integral part of many other related standards, notably XSLT and XPointer.

XPath 2.0 is a superset of XPath 1.0 and was approved as a W3C specification on January 23, 2007. XPath 2.0 supports a richer set of data types than does XPath 1.0 and is designed to take advantage of the full set of data types in XML schema. Significantly, XPath 2.0 also conforms to the XQuery/XPath Data Model (XDM), which was published as a W3C Recommendation simultaneously with XPath 2.0.

XPath 2.0 has a backwards compatibility mode to ensure that "nearly all" XPath 1.0 expressions continue to work with XPath 2.0.

## XQuery

**Status:** Adopted

XQuery is a query language for retrieving and interpreting information from XML-encoded content and data. It was designed with flexibility in mind, recognizing that XML-encoded information is a wide range of sources, including data and documents. The W3C recommendation is careful to note that XQuery operates on the logical structure of an XML document, or the data model, as defined in XDM.

XQuery is a significant addition to the family of XML standards, and by some measures, the most significant apart from XML itself. XQuery allows XML documents, Web pages, and database content alike to be searched using a powerful, XML-aware language. Individual documents, collections of documents, or nodes in a database can be searched for, selected, processed and manipulated, similar to searching capabilities found in other query technologies.

The XQuery standard consists of a number of different standards which work together to define the requirements, data model, and syntax of a data query. These include:

- XDM
- XQuery 1.0 and XPath 2.0 Formal Semantics
- XQuery 1.0 and XPath 2.0 Functions and Operators
- XSLT 2.0 and XQuery 1.0 Serialization
- XML Schema

Except for XML Schema, the other standards were all released simultaneously with XQuery 1.0 as W3C Recommendations. (XML Schema was released in October 2004.)

Even during its draft status, there was significant support for the XQuery 1.0 and XPath 2.0 specifications from vendors and solution providers. With the formal adoption of the standard, we expect more support from vendors and more implementation from users.

## DOM

**Status**: Mature

The W3C Web site describes the Document Object Model (DOM) as, "a platform- and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of documents. The document can be further processed and the results of that processing can be incorporated back into the presented page." Commonly referred to as an "API into XML Documents" the DOM enables powerful manipulation of XML document nodes for a wide range of applications.

By using the XML DOM, processing and system development can be simplified. A document can be extracted from a database and loaded into a DOM aware tool and rearranged, searched against, etc. without having to perform extensive changes to the original applications or data structures. This is particularly useful in content reuse applications and for environments that are very dynamic and require *ad hoc* access and storage of XML information.

## XPointer

**Status:** Mature

XPointer is used to extend the reach of XLink, XInclude and other XML specifications to XML resources distributed across the Internet. It uses URI references to locate these resources and the power of XPath to identify and retrieve internal data structures within them.

XPointer allows organizations to extend their reach well beyond corporate boundaries by defining XML documents that include XML data found anywhere on the Web. For example, a single Web page might link to or include information found in numerous external XML resources, such as other Web pages or applications including Web services or XML-aware databases.

The XPointer standard consists of three different standards which work together to define a data reference.

## XLink

**Status:** Mature

The XML Linking Language (XLink) allows cross reference or link elements to be inserted into XML documents that refer to documents, graphics, or other applications. XLink extends the capabilities found in HTML unidirectional linking with additional functionality, including:

1. Defining extended linking relationships among more than two resources, sometimes called "one-to-many" or "many-to-many" links,
2. Associating metadata with links, that may be used to resolve links or be used by applications at the target location,
3. Expressing links that reside in a location separate from linked resources,
4. Associating behaviors with links, such as whether to open a new window, open in the same window or to bring the linked content into the position of the current document where the linking reference resides.

The end result is a robust linking expression language that adds considerable power to XML data and applications.

Organizations can take advantage of XLink capabilities to create powerful links into databases instead of flat files on a Web site. Also, links can be defined to result in powerful behaviors such as opening a new window and including several documents as if they were maintained as a single HTML file. These links enable very powerful and dynamic document processing that avoids a lot of static Web page development. Also, the relationship of the links expressed in XLink outlive any specific instance of the data so links do not break as easily as in other linking methods.

## XInclude

**Status:** Mature

XML Inclusions (XInclude) is a language used to define the inclusion or merging of XML documents to support modularity. Expressed in a friendly XML syntax, XInclude expresses which documents or document components are to be merged and the processing behavior to be applied. The syntax leverages the XML constructs of elements and attributes, as well as URIs.

XInclude supports merging the contents of files or portions of XML documents expressed using XPointer. XInclude extends the capabilities for merging expressed in XLink by adding a processing model to control merge processing. While XML documents can be merged using several approaches, XInclude can be applied independently of other processing. Prior to XInclude support, users would have to rely on custom or proprietary processing done by a content management system or other mechanism. While such an approach might work well in a specific

environment, it did not allow ready integration or extension of the XML processing with other systems that did not use the same proprietary approach.

XInclude would be used by an organization that wants to maintain documents as modules that are assembled as needed. Perhaps highly controlled boilerplate text such as legal disclaimers would be included in many documents during product staging or delivery. This would avoid inconsistencies, or even liabilities from text errors in the resulting delivered documents.

## Areas Lacking Completed W3C Standards

As users begin using XML databases, they will be faced with some design and development issues that are not currently addressed by a consistent and open standard. In these situations, they will have to rely on proprietary capabilities of the databases they deploy. But, different databases may support the functionality in wildly differing ways, which reduces the interoperability of systems and increases development and integration efforts.

More powerful XML databases will utilize existing XML standards to support these functionalities as much as possible to reduce the interoperability challenges as much as possible. As more people build sophisticated XML database applications, the demand for standards that address the following areas should increase.

### Full Text Search in XQuery

The XQuery committee in the W3C prepared a draft set of requirements for supporting full text searching in an XML environment in May of 2003. Subsequent work includes a series of use cases that illustrate how XQuery might enhance full text search capabilities (the W3C released a paper in July of 2004). A "Working Draft" of the Recommendation was published on May 18, 2007. As with any working draft in a standards development activity, this document and the requirements it describes should be considered suggestions and are subject to change.

Full text searching is a broad range of functionality that could be subject to many interpretations. It is clear, however, that utilizing XQuery while specifying and executing full text searches would provide significant benefits to search precision.

### Updates in XQuery

XML databases have a variety of strategies for updating and deleting documents. These range from replacing or deleting the existing document, to modifications applied through a live DOM tree, to languages that specify how to modify fragments of a document. Currently most methods to accomplish this type of functionality are proprietary.

There is recent activity where people have explored the creation of standardized languages for updating XML documents. Most implementers are looking to the

W3C effort, "XQuery Update Facility 1.0," which published a working draft in August of 2007. XQuery Update is designed to create a set of extensions to XQuery to address updating XML content and to formalize them in an official standard.

## Versioning

Document versioning in an XML database involves a set of functionality that manages the relationship of document components that are shared in multiple document instances. As one version is updated, the need to decide whether related versions also need to be updated, or if the relationship needs to be continued, can be supported by the database application. Different XML databases manage these relationships and decisions to varying degrees. There are complexities, such as whether branching is allowed, that may or may not be supported by different tools.

XPath, XPointer and other XML standards can play a powerful role in defining and managing these relationships. But until a standard that addresses XML database versioning emerges, implementers will need to consider the specific capabilities of XML databases and the impact their capabilities will have on their specific development system projects.

## Collections

Most XML databases support the concept of a collection, or grouping multiple individual documents. This is similar to the way a table in a relational database or a directory in a file system groups multiple instances of records or documents.

If an application using an XML database manages invoice data, it might be very useful to the accounting department to refer to collections of invoices and structures above the individual invoice during processing sets of invoices related to a single customer. Similar relationships can be useful in groups of documents such as product descriptions, technical manuals, and many other examples. Collections might even contain other collections, depending on the capabilities of the specific database system in use.

Currently only XQuery uses the concept of collections, but it is not difficult to see how XPath, XPointer, XLink, XInclude and other standards could use this concept as well.

## Transaction Management

Transaction support is a critical feature of database systems. Wikipedia offers an excellent definition of a transaction as, "a unit of interaction with a database management system or similar system that is treated in a coherent and reliable way independent of other transactions." Software developers often view transaction support as a critical functionality in data management, and thus will often lean toward relational database management systems because of what they see as more mature transaction support. XML databases support transaction management in some form or another (including updating elements and presumably rolling back to

prior versions). However, locking is often at the level of entire documents, rather than at the level of individual nodes, so multi-user concurrency can be relatively low. Whether this is an issue depends on the specific application and what constitutes a "document". Many XML implementations rely on being able to manage information, and the transactions applied against that information, at hierarchical levels below the document level. Some need to manage their information at very granular, low level structures.

As desirable as it is for node-level locking in XML databases, there are complexities with node-level locking. Locking a node usually requires managing control of parent and child nodes, including, for instance, whether a parent can be updated or deleted while one of its children nodes is locked by another process. Schemes for defining and expressing transaction management rules are being discussed in various technical communities. But, until a standard addresses this area, implementers will need to depend on the specific capabilities of the XML Database systems they implement. These vary to a large degree from one database system to another.

## PSVI API

The XML DOM provides a means of accessing data as it is processed. It relies on the canonical form of the XML data produced by XML processors called the Infoset. When XML data is processed using a W3C XML Schema an additional infoset is created called the Post Schema Validation Infoset or PSVI.

The advantages of being able to load XML data into DOM processing tools and having a formal "API" into that form is very clear. Similarly, an API or formal syntax for accessing and manipulating data in the PSVI form would provide beneficial processing efficiencies and capabilities when processing data against a XML schema.

There are groups, such as the Apache Project, that have created PSVI "API" tools, but PSVI API abilities are likely to remain fragmented until a syntax is formally developed and passed by the W3C XML Schema working group.

# Customer Use Examples

## Importance of XML to Users

The XML family of standards has become a critical component to many content publishing and distribution systems for users of all types. XML has enjoyed popularity for single source publishing due to the benefits of reusable content. Even so, it is the combined capabilities of the set of XML-related standards and the XML databases that support them that enable a uniquely powerful platform for managing even the most complex and demanding content applications.

> ***The interoperability and flexibility provided by software taking advantage of the benefits of XML standards positions users to deliver significant competitive advantage to their organizations and those of their customers and partners.***

This section describes two examples of how databases that implement XML standards deliver powerful benefits to content creation and delivery.

### An Airline Company

The major goal of an airline is to provide air travel service to an audience in a highly competitive market where costs and efficiency can make the difference between success and failure of a business. Aircraft are expensive and complicated, and the information used to maintain and operate them is uniquely complex. Each aircraft in a fleet may have different specific components, and the information used to maintain and repair this equipment needs to be accurate and current.

The airline industry is extremely concerned with safety and must modify existing manuals and work cards to keep them current with aircraft manufacturer and government regulatory information as it is made available. Finally, a large airline has users of this information distributed all over the world, and providing timely access to the correct information from anywhere in their systems is a challenge faced by few industries.

Airline maintenance information is organized into technical manuals and task oriented work cards, as well as many other derivative information formats. Technical manuals are very comprehensive, while work cards are organized for a specific maintenance or repair procedure that can be accomplished by a specialized worker in a single shift. They may be made of much of the same information but are organized in very different formats. Even so, the relationships of the information components that comprise them need to be maintained so they are current, synchronized, and highly accurate.

Traditionally, airlines have managed their maintenance information in systems that produce books and documents. The current maturity of XML standards and the support of these standards by XML databases make it possible to break out of

document-centric publishing into an environment where the information is highly integrated and easily reused. Data coming from diverse sources, manufacturers, internal departments, and regulatory agencies can be quickly integrated and inserted into information objects that make up the diverse delivery formats. The relationship between a repair step in a work card or manual and the government directive that changes that procedure is quickly identified and executed upon.

The airline industry has been using XML and SGML technology for many years for sharing information. Newer equipment in airline fleets may be captured and managed in the industry's ATA 2200 specifications, but documentation for older aircraft may still reside in paper form, or older proprietary publishing system formats. Also, the data contained in these documents must be delivered in a variety of consumable forms, including print and the Web.

Given these challenges, most airlines are looking to create a single repository of maintenance information components to facilitate effective reuse and repurposing. The end result is to improve accuracy and efficiency, while reducing costs and delivery schedules.

The links between information objects fall into three categories:

1. In some cases, when an information component that exists in one document but is also used in several others is changed, the new content should automatically be replaced in all locations.

2. In other cases, there are significant differences in the actual text in two related objects, so when information is changed in one place, the system must track that the information needs to be manually updated to be kept in synch,

3. Lastly, in some information objects, sub-components can be automatically updated, while others need to be manually updated.

This challenging reuse model is difficult to maintain using manual tracking methods. XML standards for structure and relationship linking supported in EMC Documentum XML Store enable these processes and updates to be significantly automated or at least tracked and managed by the system. Updates to a technical manual that comes from an aircraft manufacturer supplier may trigger several automatic updates, initiate several other manual workflow steps, generate multiple delivery formats such as HTML, PDF and others. Also, it may initiate updating of management information to control the process and report on steps taken for tracking and compliance purposes.

Airlines, like many other large corporations, are large operations with diverse user and authoring roles, and a heterogeneous mix of software, operating systems and data formats. Authors tend to be highly concentrated in one location on a consistent set of workstations. Meanwhile data and application servers might be on other platforms that host applications and databases. Finally, consumers of this

information are highly distributed and need to work in a mixture of paper documents and Web clients. The XML standards supported by EMC Documentum XML Store enable data that can be captured, edited, and rendered throughout a heterogeneous environment such as this.

For these reasons airlines are interested in avoiding being locked into a proprietary format that would insist on more consistent environments for all phases of document and information production and consumption. Also, considering that most aircraft equipment, the airplanes themselves, remains in service for 20 to 30 years, the data used to support them must remain portable and outlive the environment in which it is processed today.

According to one airline industry executive who manages these types of data, "There is no other technology that provides what XML standards do to meet these requirements."

## A Manufacturing Company

Information about products is a critical feature of customer satisfaction to any manufacturer. When a product is fairly complicated, such as appliances, machinery, or even vehicles, the supporting information is proportionately sophisticated. Managing that information in a cost effective manner, according to government safety regulations, and in a form that enables all required delivery formats is an essential part of the overall manufacturing process.

A manufacturer of recreational vehicles needs to produce the user manuals that customers rely on from the same information that dealers and repair shops use to inform potential buyers of features and specifications and to maintain the equipment once it is purchased. The technical documentation for these personal vehicles is subject to content management challenges.

For example, one manufacturer of recreational vehicles maintains documentation on 28 different models that may include one or more of 400 separate alerts (shared warning information). Also, they produce more than 3,000 components that can be purchased separately and installed on the vehicles, and must provide installation instructions for each of these as stand-alone documents or as parts of other documents. In all it is a complex data management puzzle.

Components on these vehicles can be shared between different models, so the documentation on these vehicles must be similarly shared and tracked. These manuals, alerts, and product specification sheets are made available in printed form for customers, as well as electronic formats for dealers and internal staff. Many safety regulations and standardized language rules affects how these information objects are reused as well.

In selecting a system to manage its documentation, this manufacturer wanted to take advantage of the XML structure and processing standards as well as best-of-breed document component management functionality found in EMC Documentum

XML Store. The focus was on reducing the costs and complexity of managing the relationships between information objects and their reuse in the many specific documents and output formats.

Challenges included looking for technology that was compatible with existing systems and workflows. The integration capabilities expected from EMC Documentum XML Store, due to its support for many XML-based standards and industry APIs, made it a logical choice for the data management system for their documents. The XML standards allowed documents to be created and managed so that they could be reused in different manuals and repurposed for different output formats.

New and existing processing environments may utilize many third-party tools for various processes. The industry APIs supported by EMC Documentum XML Store enable integration with existing systems and the existing efficient workflow processes, such as structured text editors, composition systems, and Web content management tools.

In the resulting environment based on EMC Documentum XML Store, searches utilize XQuery and are very precise. XQuery provided better search results than some of the other XML tools and technologies considered. EMC Documentum XML Store, through XQuery, enables document components to be shared reliably and quickly. EMC Documentum XML Store also supports the WebDAV architecture which provides stateless editing, powerful file locking controls, and enables applications to work in the heterogeneous legacy environment.

Due to the high degree of document component sharing and reuse, strong versioning capabilities of EMC Documentum XML Store enabled warnings and cautions to be tightly controlled and rules enforced about which version is in use at any point in time. Warnings and cautions come from internal engineering and technical writing personnel, as well as regulatory sources, and are a critical part of the documentation.

Many manufacturers, such as the one described above, enjoy sales of their products to countries around the world and need to produce supporting information in multiple languages. One significant benefit of publishing documents using systems that take advantage of XML and related standards is that translations into different languages have become more efficient due to the strong information object sharing. Because a single object is reused it only needs to be translated once for each language. Previously, objects were duplicated and maintained in multiple files and usually became slightly inconsistent over time. XML-aware version control and transformation prevented the relationships of different versions from becoming lost and reduced the need for much of the translation due to minor textual differences.

This manufacturer did not want to sacrifice performance to get the benefits of robust information sharing and automated content management and processing. The power of XML standards based information processing is very apparent in this publishing application.

# Conclusions

Organizations intent on creating powerful and efficient environments for managing and accessing XML data have found that traditional relational databases are not designed to work with hierarchical XML content. Excessive application development and inadequate performance and efficiency result when tools that take advantage of XML content and standards for processing are not deployed.

A new breed of tools called XML databases not only understands the structure and syntax of XML content, but also utilizes the powerful standards developed by the W3C to process and access XML data. The W3C, developers of XML, have produced many related standards, including specifications that provide XML processing and database support. These include XML DTDs & Schemas, XSLT, DOM, XPath, XPointer, XLink, and XQuery. These specifications are well tested and deployed by organizations managing complex XML content. They enable a powerful platform for managing XML data that surpasses alternative traditional approaches for data management.

> ***Developers seeking to build a best-of-breed XML content management environment need to understand the capabilities that XML databases bring that cannot be achieved using non-XML aware tools.***

Information-intensive businesses such as an airline and a manufacturer have built applications that demonstrate the power and efficiency of using an XML database such as EMC Documentum XML Store to manage their XML information assets.

With Documentum, EMC provides an enterprise-class content management system with a unified content architecture. The Documentum platform provides a unified environment for storing, accessing, organizing, controlling, retrieving, and delivering any type of unstructured information within an enterprise.

As a part of Documentum's next release, XML Store can be added on to the Documentum Content Server and will allow XML document storage and access via XQuery. The documents stored into XML Store will be part of the native Documentum repository, accessible through standard Documentum APIs and query language, and subject to the same policies and management as other documents. Customers can now get both the security, compliance and archiving capabilities of the Documentum platform and the high performance provided by XML Store.

# Resources

W3C Activities, World Wide Web Consortium (W3C), http://w3.org/Consortium/Activities

"Going native: Use cases for Native XML Databases," R. Bourret, http://www.rpbourret.com/xml/UseCases.htm

EMC Documentum XML Store, http://www.emc.com

# Sponsoring Company Information

For more information, please contact:

EMC Corporation
176 South Street
Hopkinton, 01748
United States
Phone: +1-866-464-7381

On The Web:
http://emc.com