Session 7: Information Modeling with XML

It may sound difficult to choose between Object-Role Modeling (ORM), and the Unified Modeling Language (UML). UML and ORM are obviously modeling technologies, but where does XML fit into the picture? Well, XML is more than a document markup language – it is a solution for content modeling and creating standards for content. three types of modeling that will be important technologies in the new millennium.

All three types of modeling are important technologies today. This trinity of modeling techniques includes object modeling, content modeling, and data modeling, using the Unified Modeling Language (UML), the eXtensible Markup Language (XML), and Object-Role Modeling (ORM), respectively. To build some Web applications, you might use two or three modeling techniques. For example, there are e-commerce working groups using UML and XML to create new electronic data interchange (EDI) standards. Developers creating multilayer applications might use UML for middle-tier components and ORM to produce database schemas. UML and ORM share concepts such as an object-centric methodology and conceptual modeling, but ORM is a tool for designing databases, not application classes.

Developers have been building information systems or information engineering models for several decades. The first generations of modeling tools and technologies centered on process and data modeling. System-level flow charts are early examples of process modeling, whereas Natural Language Information Analysis Method (NIAM) and data flow diagrams (DFD) were early data-modeling techniques. More recent modeling technologies support object-oriented analysis and design (OOAD). Data modeling is the process of creating models when designing databases, typically following a progression from conceptual model to logical model to physical schema. Most data-modeling products support entity-relationship diagrams (ERD), object-role modeling (ORM), and Integration Definition for Information Modeling (IDEF1X) models.

Because developing systems is a challenge, the software industry has produced a variety of tools to automate analysis and design. None has gained wider support than the UML adopted by the Object Management Group. The UML is an object-oriented methodology that combines separate work by three well-known experts: Grady Booch, Ivar Jacobson, and Jim Rumbaugh. It standardizes modeling language and notation, not a particular method. UML supports several different views of a system -- class diagrams, behavior diagrams, use-case diagrams, and implementation diagrams. Use-case diagrams let UML users define how actors participate in an interaction with the system. UML users can capture the system's dynamic behavior by using activity diagrams, collaboration diagrams, sequence diagrams, and state diagrams. To document
the lower-level details of a system, UML users can develop component diagrams and deployment diagrams. The UML is also extensible to support new modeling concepts by the use of stereotypes and patterns. The UML is a powerful solution for application object modeling, but developers doing realtime and database applications find it doesn't address all of their needs. For those purposes, developers often turn to data modeling and ORM.

To model with ORM you can use Visio Enterprise, a product that also supports UML. If you're building a multitier application, you can use ORM for the database tier and UML for middle-tier objects or components. Of specific interest is Visio Enterprise's ORM tool known as VisioModeler.

The chart below represents an ORM conceptual model for a memo-style document. The model can be used to generate a database schema for storing documents.

![ORM Sample Conceptual Model](image)

Figure 1 – ORM Sample Conceptual Model

The proposed chart corresponds to an ORM conceptual model of a memorandum modeled with Visio Enterprise. Developers working on the database tier of an application start with conceptual modeling when designing database schemas and generating SQL Data Definition Language (DDL) statements.
Developers working on other tiers of an application use object modeling tools to design components or classes used by applications. The most popular tools for modeling application objects support the Unified Modeling Language (UML). Whether creating databases or application objects, modeling lets you apply formal techniques, including design-time validation, before you build a physical database or application objects. For a rigorous design, software professionals might use ORM for databases and UML for components.

**The Value of Modeling**

Why model? The short answer is quality. Developers face a challenge when developing complex databases or applications having many objects. For example, if you're building an e-commerce Web site, you're likely to create models that include objects such as customers, orders, payments, shipments, invoices, and so on. As you add more functionality to your Web applications, you add more complexity to databases and software. Decades of software quality-assurance research have revealed certain axioms. A major finding was that most application flaws and bugs are the result of a deficient design. Studies have shown that the expense of correcting errors increases as a project progresses through the development and deployment cycle. In other words, it is less expensive to correct an error during design than during coding, testing, or deployment. The message for developers is: design, design, design. The Federal Aviation Administration's highly touted Bandwidth Information System used a design phase whose duration was about 50 percent of the project schedule.

XML promises to be a major influence on software quality. Developers today can use object models, database models, and standard components. As XML gains momentum, developers will also use document type definitions (DTDs) as the content model for standard inputs and outputs. Every generation of developers has its magic silver bullets -- technologies that will solve all of our software problems. Java and XML have attained that status, in part because of the zeal of Web developers who favor vendor-neutral software technologies. Java and XML are not a panacea, but they do promise to simplify development using standard components. They're important ingredients in a recipe that should include formal development methods such as modeling. The purpose of this handout is to describe how XML can be used for data and object modeling.

**Describing Content, Describing Data**

The industry is devoting a lot of energy to describing data, modeling information, and developing standards for metadata. The interest in XML is exploding and the largest software companies are in a race to provide XML-enabled products. There is a flurry of activity to develop DTDs for use with XML-enabled software.

XML lets you model content that ranges from book-oriented document types to commercial transactions. It's an enabling technology for business-to-business integration, data interchange, e-commerce, and the creation of application-specific vocabularies. Database developers are particularly interested in XML as a building block for creating middle-tier servers that integrate data from disparate databases. XML is also used for purposes such as exchanging data warehouse
metadata and UML models. Organizations such as IBM, Oracle, Rational, Sybase, and Unisys are promoting XML, UML, and the Meta Object Facility (MOF) as solutions for collaborative development using an open information interchange model. MOF is an Object Management Group (OMG) standard for distributed repositories and metadata management. The marriage of XML, UML, and MOF produced the XML Metadata Interchange (XMI) specification.

**E-Commerce Is Driving Content Standards**

The growth projections for e-commerce are believable when you consider the flurry of development and standards activity, as well as the number of organizations building e-commerce infrastructures. There is a potential for business opportunities if you comply with e-commerce standards, and a potential for lost business if you don't. Plugging into the government's e-commerce network should streamline business processes and reduce the cost of doing business with the government. Not developing XML expertise means you will be unable to fulfill documentation requirements or cut costs with e-transactions. There is other evidence of the importance of e-transactions and XML capabilities. The Debt Collection Improvement Act (DCIA) requires, as of January 2, 1999, that payments by the U.S. government (except tax refunds) be made by electronic funds transfer. In April 1999, the state of Utah adopted XML documents as a standard for court filings. One of your requirements will be delivering technical documents that conform to standard DTDs.

The BizTalk initiative is another reason for organizations to become familiar with DTDs and XML. BizTalk is a Microsoft effort to create a new XML-based commerce architecture. Microsoft and its partners are developing standard e-commerce DTDs that support activities such as exchanging catalogs and corporate purchasing over the Internet. The partners include 1-800-FLOWERS, Active Software, barnesandnoble.com, Claris, Commerce One, Dell Computer, Eddie Bauer, Harbinger Corporation, J.D. Edwards & Company, Level 8 Systems, MasterCard International, Oberon Software, PeopleSoft, SAP AG, and Sharp Electronics Corporation.

Because XML and Java are inexorably linked as "hot" Web technologies, there's a lot of Java code being written to produce or consume XML. For example, companies (IBM, Microsoft, Oracle, Sun) and individuals (James Clark, Tim Bray) have published XML parsers written in Java. Sun's Java Project X lead to provide technology that supports XML with Java classes. The current releases of Enterprise JavaBeans use XML for deployment descriptors and versioning information. IBM has made a huge investment in XML and Java, and the Apache’s XercesJ which was derived from IBM’s XML4J is one of the best XMLparser available today. Not to be outdone in the XML department, Microsoft fully supports XML in Office 2000 and Internet Explorer 5.0.

**What Does XML Let Us Model?**

XML lets developers design application-specific vocabularies. Its popularity is based in part on human readability and ease of understanding.

```xml
<?xml version="1.0"?>
```
Web Farming for the Data Warehouse (Morgan Kaufmann) is a compendium of information that is best-described as systematically making intelligent use of the Web.

The book contains a wealth of information about content-providers, protocols, standards, tools, discovery services, knowledge management, Web agents, and data mining software.

Hackathorn's book is as close to leisure reading as I expect to find in an IT book. It belongs on the must-read list of anyone having an interest in exploring the Web's potential as an information resource.

The listing above is the XML counterpart to the ORM model of Figure 1. XML enables you to describe structure, but it doesn't define presentation rules. Some people see XML as document-processing technology, but it's also a powerful tool for database projects. A visual designer thinks of XML in terms of client-side issues such as presentation and style sheets. A database developer is more likely to be concerned with middle-tier data integration and schemas. To effectively use XML, a database developer needs to understand its structure and how to store it, parse it, manipulate it, and generate XML output from database queries.

XML defines content standards through the use of industry-accepted DTDs, although the DTD is optional in XML documents. The DTD defines rules and structure for a class of documents. You can use the DTD, for example, to define tags that are required in a document. You can model a document's structure by using tags that define document elements, entities, and attributes. Entities represent a document's physical structure and elements represent its logical structure. Attribute lists are used to specify element metadata, such as unique identifiers.

XML documents contain information that conforms to a tree structure. A simple XML parser checks a document to see if it's syntactically correct, and a validating parser verifies that it's semantically correct (that it conforms to the DTD). There are a number of free XML parsers on
the Web written in Java, C, and C++. The W3C has developed a Document Object Model (DOM) that abstracts and layers objects over the XML document structure. The DOM lets a developer program with standard objects while manipulating XML data trees.

**Schemas, XML-Data, RDF, DCD**

There are several initiatives related to XML schemas. The Resource Description Framework (RDF) provides the ability to describe Web resources. An RDF schema is similar to a database schema. It defines types and document semantics, which makes it possible to do type checking of XML documents. Schemas also make it easier to build query processors. Dr. Neel Sundaresan of IBM Almaden Research Center coauthored the W3C proposal for an RDF Query Language. The Document Content Description (DCD) specification describes a solution for creating structural schemas or document content format descriptions. DCD lets you define rules that describe the content and structure of XML documents. DCDs supplement DTD semantics by describing constraints in XML and defining document elements as types. DCD constraints include subclassing and inheritance, data types, and SQL-style constraints (key fields, unique values, referential integrity). XML-Data is an XML vocabulary for defining and documenting object classes. It can be used to define the characteristics of syntactic classes, or for conceptual schemas that are similar to relational schemas. XML-Data schemas include element type declarations. They support subtypes, groups (ordered sequences of elements), defaults, aliases, class hierarchies, multipart keys, and one-to-many relations. W3C recently released the XML Schema recommendation that provides an integrated set of document structuring and data typing capabilities.

**Queries, Events, Linking**

One of the reasons for defining XML schemas is to let software developers build sophisticated tools for querying XML data stores. XSL contains sophisticated pattern-matching functions, but it doesn't provide create, update, and delete operations that are analogous to SQL queries. When calling for input on query processing for XML, the W3C received dozens of position papers.

Developers writing software such as Windows applications or Enterprise JavaBeans write code that uses an event-processing model. When writing a typical Windows application, for example, you provide event handlers for different Windows messages. The XML standard doesn't define events, but David Megginson developed the Simple API for XML (SAX), a solution that lets Java programmers associate event handlers with tags. XLink is an XML linking language specification being reviewed by the W3C. XML supports external file references but it currently doesn't include predefined element types (such as `<IMG>`) that define a hypertext-style link. Today you can use XML to create content tags (`<para>` or `<city>`) but there is no way to identify hypertext link semantics for those elements. XLink is a proposal for extending XML with features such as bidirectional links. XLink reserves attribute names, such as `xml:link`, to signal a hypertext link.

XPointer is a spec for an abstract language used for specifying locations in documents.

```xml
<A HREF="http://www.xyz.com#ID (foo)CHILD(4,SEC)CHILD(1, ABSTRACT)"
```
XLink and XPointer are works in progress but some developers have implemented software based on the working drafts.

**Databases, Tree-Structured Data**

Database developers are likely to be involved with XML for several reasons. You might develop applications that query databases and format the query results as XML documents. You might have to develop databases and write code to support business-to-business integration or other forms of electronic data interchange (EDI). You might also have to store XML documents in a database. Today's object-relational DBMS products from Oracle, Informix, and IBM support text databases and sophisticated text searching. It's no surprise they are developing XML-centric extensions to their product lines.

Some XML advocates have asserted that XML and databases are divergent technologies. This assertion is based in part on an erroneous assumption that a DBMS can manage row and column data, but not tree-structured data. Database veterans recognize that hierarchical and Conference on Data Systems Languages (CODASYL) model databases have no problem with hierarchies or trees. Those are legacy technologies, but object and object-relational databases also store text data and operate with tree structures.

SQL-92 also offers solutions for tree-structured data, as discussed in Joe Celko's *SQL for Smarties* (Morgan Kaufmann). Parts explosions and bill of materials processors (BOMP) are examples of applications where developers use SQL with inherently hierarchical data. In the late 1980s, Dutch researchers used NIAM to create a conceptual model and Oracle document database that was the semantic equivalent of SGML. The basic process of designing SQL databases often involves decomposing an object or entities into its constituent parts. To store an XML document in a SQL database, you can follow that paradigm, although some products can store the document as a single column.

Oracle has built support for XML into products such as Oracle Application Server and Oracle 8I (O8i), an object-relational DBMS capable of storing rich data types. O8i includes an XML parser, an Internet File System (iFS) that provides automatic parsing of XML documents, and Java classes that support XML. Oracle's parser supports the Document Object Model and the event-based SAX interface. Oracle's interMedia text search engine provides section searching of XML documents.

IBM added an XML toolkit to its WebSphere (application server) studio product and added XML capabilities to DB2 Universal Database (UDB). It has developed an XML Extender for DB2 UDB and has added features for storing and querying XML documents. It can store XML documents as a single column, or decompose the document and treat it as a collection of fields that are standard or user-defined types (UDTs). IBM revised the DB2 text search engine so it understands the structure of an XML document, thereby providing capabilities such as section searches. By storing XML elements and attributes as SQL types, DB2 users are able to index documents for more powerful search optimizations. IBM also added functions to extract XML elements, attributes, or entire documents, to reconstruct decomposed documents, and to link to XML documents stored in external files.
Microsoft made a heavy commitment to XML. Office 2000 saves files as native HTML and includes embedded XML within Office documents. Internet Explorer 5.0 (IE5) can read XML tags embedded in a Web page and perform semantic validation using DTDs. Microsoft implements the W3C DOM over a Component Object Model (COM) component. Developers can use the XML COM component for client- or server-side programming. The XML component calls the parser (processor) to check a file for validity, builds the document tree in memory, and then builds the object model over the tree. This programmatic interface enables VBScript, JScript, ActiveX Data Objects (ADO), and Active Server Pages (ASP) to operate with XML. A script can, for example, create an XML data source object (XMLDSO) and then use DOM objects to walk an XML tree. ADO has been extended to work with XML as shaped or hierarchical recordsets. Users can bind data islands to Web pages and then use XML objects or ADO methods to navigate through the data.

**Understanding the Example Model**

In the example conceptual model in Figure 1, the primary object is an email message, interoffice memo, or similar communiqué. This corresponds to the ELEMENT tag in an XML/SGML DTD. Content modeling, object modeling, and data modeling are useful technologies that contribute to rigorous design and development activities. They promote component reuse and make it possible to use a cookbook approach to developing systems. In this handout, we've touched on content modeling with XML and data modeling with ORM. Both relate closely to object modeling with UML.

**Useful References on XML Information Modeling**

BizTalk
www.microsoft.com/PressPass/commerce/initiatives

Comparing RDF-Schema and UML
www.w3.org/TR/NOTE-rdf-uml

Comparison of Data Modeling Techniques
essentialstrategies.com/documents/comparison.pdf

GUIDE Business Rules Project Report
www.guide.org/ap/apbrall.htm

GUIDE Business Rules Project Report (Adobe PDF)
www.guide.org/ap/apbrules.pdf

Microsoft XML Site
www.microsoft.com/xml
STEP tools
www.stepools.com/projects/niip

Query Languages proposed for XML
www.w3.org/TandS/QL/QL98/pp.html

Visio templates for HP Fusion
www.hpl.hp.com/fusion-mf_visio.html

W3C Metadata Project
www.w3.org/Metadata/Activity

XMI
www.oasis-open.org/cover/xml.html

XPointer and XLink implementations