XML provides the context for turning a data set into information; KM provides the tools for managing this information.

A n appreciation of how context turns data into information is fundamental to understanding XML’s role in a KM infrastructure. Here, “context” refers to the identity of, and relationships between, a data set’s entities.

Information exists when data and context combine to associate data with specific concepts. For example, associating 72 inches with a concept such as a person’s height provides the context that turns this data into information.

In other words, information is meaningful data; data without context is meaningless. Implementing a database’s schema by specifying the table structures provides the data context. Populating the database brings the data and context together to make information.

Customers entering data into a Web form’s text boxes have a mental model of the data they’re using. They associate a particular meaning to terms such as “first name,” “last name,” and “address,” and they expect these data entities to be associated with individuals.

When deciding what data to put in each text box, customers match their mental model with the model that the text boxes’ names and layout represent.

The hypertext transfer protocol keeps each text box name associated with the data while it is being sent over the Internet to the database server.

At the server, the database management system uses that text box name to determine which table and column to place each data entry in, thus keeping the data associated with that name. The database schema sets the rules for this association between the data and the name. These rules must be consistent with the Web form’s presentation.

Thus, the data entry Web form serves as an interface filter to ensure that the data and context translate properly from the customers’ mental model to the schema’s data model.

Customers see the data as information because it is in a context they can associate with a mental data model. The database entry process preserves that...
context, passing the information from the mental data model to the schema data model.

Designers of Web forms and server database input processes must ensure that this translation is accurate so that the two models are equivalent.

Data processing’s role in general is to control the movement of data from one context to another so that it continues to represent information. If it ever gets out of control and loses its context (for example, if a database is corrupted or the Web form is unclear), the data no longer represents information, because it loses its meaning.

So where does knowledge come into play? Just as information is the union of data and context, knowledge is the union of information and specific issues. Therefore, we can define knowledge management as aggregating data with context for a specific purpose.

**DTD STEALS THE SHOW**

The conventional approach to knowledge management keeps data in a database, relying on the database schema to provide the context. However, this approach is becoming awkward as the KM domains scale to potentially thousands of databases.

Writing an IT application that uses data requires having knowledge of that particular database’s schema. An application that interacts with N databases must interface with up to N different schemas. Open standard protocols such as Open Database Connectivity standardize only the connection to the databases. The programmer must still deal with each particular database’s schema.

In addition, as the Web’s popularity as a standard vehicle for information delivery continues to grow, HTML’s inadequacies are becoming a significant problem. HTML lacks the flexibility to efficiently represent anything beyond a data model’s presentation aspects. Thus, information transfer depends on the programmer’s ability to create a descriptive data context presentation and the user’s ability to properly interpret it. This process is extremely inefficient and error prone.

We can learn a useful lesson from manufacturing. Years ago, engineers recognized the inefficiency of having a robot in one automation island throw each part it created into a bin where another robot retrieved it. Throwing the part in the bin needlessly lost information about the part’s orientation. The solution was to place the part on a table or conveyor belt so that it retained its orientation.

In KM, writing special routines when an application must connect to an information source via the Web is analogous to having the second robot do the bin-picking task. Once again, the solution is to preserve the orientation (data context) as the part (data) goes from one automation island (knowledge repository or lab bench) to another.

A global XML DTD, along with the tools necessary to incorporate it into a corporate KM infrastructure, is the key to this solution. XML can also greatly improve the accuracy of information transfers and open up opportunities for total automation of manual KM processes.

**XML MAKES ITS MARK BEHIND THE SCENES**

Discipline and tools are the two key factors for successfully incorporating XML into a corporate KM infrastructure. Participants must understand, accept, and properly use the part of the data model that impacts their activities. This requires developing a consensus for creating the initial data model.
model and carefully monitoring to maintain that consensus. The data model design must provide an efficient means for incorporating future developments in the KM infrastructure.

Successfully incorporating XML into a KM infrastructure requires that all community members participate in developing a standardized domain data model. The design must provide an efficient means for incorporating future developments in the KM infrastructure. Data modelers create an XML DTD that represents this domain data model. Programmers, in turn, provide KM tools with reusable XML parser code so that users can easily interact with the model.

Community members use the KM tools to pass information to one another or to move it between knowledge repositories. The KM tools keep data and context together by tagging the exported data with XML syntax or by interpreting the imported tagged data according to the domain DTD. Knowledge repositories store the data either as marked-up flat files or in databases with schemas that are consistent with the domain DTD. The data is never separated from its context, and the context always accurately represents the domain data model.

Figure 1 illustrates this process. Only the data analysts and programmers directly involved in maintaining the DTD data model need to understand XML and DTD syntax. Even the programmers creating the KM tools don’t need to know X M L syntax. They need only to know how to interface with the standard XML parsers that make that data model available to the software.

This process is analogous to HTML tagging, the current presentation standard. Authors make certain presentation choices in their word processing program, such as font or heading style, and then save their documents in HTML format. Neither the author nor anyone who reads the document needs to know HTML syntax, and any browser can read and interpret the document for display.

The paradigm for using XML-aware tools is similar, but the focus is on conforming to the domain data model. For example, to expedite mining raw data in future years, a company can develop a domain data model component that captures all laboratory test results immediately at the source as XML markup. This model could include the following organizational concepts and data items:

- Test description. Corporate identifiers would include ID number, charge number, start date, end date, title, abstract, background, approach, schedule, and financial data.
- Test environment. Environmental attributes would include temperature, time, humidity, wind speed, and location. Equipment attributes would include type, manufacturer, and serial number.
- Test reading. Observation parameters would include instrument, location, time, and readings.

Fitting the raw laboratory test results to this data model at the lab bench requires augmenting existing automated data acquisition tools to make them XML-aware. Discipline is important for setting up the domain data model, creating new or modifying existing data collection systems, and enforcing their use. However, this effort can tremendously improve a company’s ability to manage its aggregated information.

XML is far more than the next evolution of HTML. Today it is expanding the Web’s potential for communicating information. As the advantages of standardized data models are realized, a longer-term and perhaps more significant impact will be improved efficiency in KM.

James H. Cook is a senior science advisor at IIT Research Institute. Contact him at jcook@iitri.org.

Resources