

Creating a Common Language: Dell's Transition to XML

By Paul Laster

Extensible Markup Language (XML) is emerging as a leading Web language, offering levels of data versatility beyond HyperText Markup Language (HTML). Dell recently transitioned its multimillion-dollar Web site from HTML to XML. This article discusses the factors that prompted Dell's decision and some of the steps involved in the transition.

Every Web language has enjoyed the prestige of being the “language of the future” at one time or another. The creation of HyperText Markup Language (HTML) was proclaimed as the start of a new generation of programming that would help build the World Wide Web as an essential tool of business and daily life. In the time since HTML first strolled onto the scene, new advancements have been made that were both innovative and misguided.

Virtual Reality Markup Language (VRML) was another step in the evolutionary path of Web design languages that became shelved by processor and modem speeds that could not meet the demands of this new technology. Extensible Markup Language (XML) also could have been one of these evolutionary dead ends, but for now it appears to promise more. At Dell, XML was used in the latest design of its multimillion-dollar Web site, and it looks like XML is here to stay.

So why XML? How could this language succeed where others have failed? Several key differences set XML apart from the myriad of other Web programming languages.

The Bottom Line is Data

XML was developed to create a way for different machines to communicate data to each other, without regard for the

presentation of that data. With XML, the data is the bottom line. Presentation is dictated later.

For example, in HTML the sentence “Bob walked down the street” could be presented in this manner:

```
<p><font face="Arial" size="12" color=
"#FFFFFF">Bob walked down the street.</font>
</p>
```

But with XML, the data content of the sentence is the primary consideration. For example, the same sentence in XML could be presented like this:

```
<sentence><subject>Bob</subject><verb>walked
</verb><predicate>down the street</predicate>
</sentence>
```

Presentation Awaits its Turn

Initially, XML would not be concerned with how a sentence looks, but rather with what a sentence contains—the data of a sentence. The presentation of XML is dictated later through the use of an XML schema, something similar to an HTML style sheet.

XML was developed to create a way for different machines to communicate data to each other, without regard for the presentation of that data.

The XML schema for the above example might dictate that each item identified as a “sentence” must have something else called “subject,” something called “verb,” and something called “predicate.” Later, an XML style sheet would merge with the XML code to decide how this content should appear in a Web-based environment.

The XML Schema Determines Form

XML schemas are referenced at the top of XML pages and are

utilized when XML code is compiled into its final output format. The schema for the above example could dictate that, when presented, a sentence must be built using subjects, verbs, and predicates in the correct order. For our example, that order is subject, verb, and predicate.

The schema could even go further to state that if the user language is German, this order should be changed. To take our example further, from a grammatical perspective, the English sentence above with German word order would read, “Bob has down the street walked.” This sentence might be understandable to someone who knows a little about German and English, but between the logic of two machines, different issues could arise because of this change in structure. The word order of the sentence might cause a problem with translation in an HTML environment, but this change in word order could be accomplished dynamically by XML.

The HTML code for this page might appear as follows:

```
<p><font face="Arial" size="12" color="#FFFFFF">Bob has down the street walked.</font></p>
```

This difference in order could be accounted for in XML. Therefore, the XML code for this new sentence would still appear in its original form:

```
<sentence><subject>Bob</subject><verb>walked</verb><predicate>down the street</predicate></sentence>
```

In XML, an XML schema could provide different options for different languages. The general makeup of a sentence might remain the same, with subject, verb, and predicate, but the order could be changed depending on the language requirements of the user. So in this example, the sentence

“Bob walked down the street” would appear in English word order as subject, verb, predicate, but in German the order could change to become subject, predicate, verb.

End Users Dictate Data Labeling

Of paramount importance in XML programming is the use of well-thought-out terms to label data. A car, for example, has many different aspects, such as tires, a chassis, axles, an engine, and so on. When designing an XML schema, it is important to decide which of these concepts—either some or all—will create the dataset that defines “car.”

This consideration is especially relevant for data that is to be shared across systems or different locations. For the example of the car, if that vehicle data is to be used by a car manufacturer, it is extremely helpful if the information remains consistent from manufacturer to distributor to dealer, and so on. In this manner, a car manufacturer may transfer all of its information about its current vehicles from a database to a Web site, then use that site as a device for distributors or dealerships to order their desired vehicles.

Because XML can port to many different outputs, this transfer of information is highly versatile. Since XML provides a method to handle data and the presentation of that data is dictated later, the same XML data used to populate the Web site for the car manufacturer could also be used to create a

Web site for car dealers.

Of paramount importance in XML programming is the use of well-thought-out terms to label data.

Furthermore, this XML data can be ported into another format to create customer information sheets or even window stickers.

Style Sheets Determine the Display

Like HTML, different style sheets can be applied to XML to display the data in different ways. Since XML uses schemas to dictate the data elements in XML code, it uses style sheets to shape the way the code looks in its output format. These style sheets affect the way the

different elements of the XML appear when displayed.

For example, if a sentence contains its three different elements (subject, verb, and predicate), the style sheet can determine how the elements of the sentence will appear. In the sentence example above, in HTML the actual HTML code is used to create the font, size, and color requirements of the sentence. In XML, this is all done apart from the actual XML code.

An XML style sheet can decide that any element of a sentence labeled “subject” must be “bold, Arial, size 12.” With this functionality, any text on the page that is “subject”

will appear in output as “bold, Arial, size 12.” The obvious benefit is that if a certain piece of data needs to be displayed differently, it can be changed in one place that will affect all occurrences of that data.

Different kinds of style sheets can be referenced for the different ways that the data is to be displayed. In the car example mentioned above, another style sheet could be used to create different output for a Web site or a window sticker. XML provides the data, the schema provides the form, and the style sheet provides a way for the data to be displayed.

Server-Side Code, Client-Side Output

Like Microsoft Active Server Pages (ASP), XML is a server-side code format. The majority of XML coding, including the actual XML, the XML schemas, and style sheets, are invisible to the end user. On an XML Web site, for example, the code visible to Web site visitors is the code type to which XML is ported. If the output of the XML code is HTML, this coding will appear when visitors to the Web site attempt to view the source of a Web page. The code will reference any style sheets in the normal HTML manner, and all of the code on the page will appear in a standard HTML format.

Why XML for Dell?

Dell decided to move its external Web site to an XML-based format for a host of different reasons. The choice to move to XML was more than just a decision based on the latest trends in Web development: XML coding provides many different options that Dell found desirable for its Web site.

Information can be Adjusted for Different Customers

Foremost, the Dell Web site is primarily an e-commerce site. Because it caters to many different customers, information on the site must be relevant to meet the needs of general Dell customers as well as those who need specific information.

Some information is the same regardless of where it exists on the site. For example, headers and footers, and other visual elements that are found on the site are relatively consistent, with some minor changes depending on their specific location. These elements can be captured in XML and then subtly changed, depending on the requirements for specific pages to meet the needs of particular customers.

Another benefit of XML is illustrated in the sentence example above. Because Dell

*XML schemas
are referenced at
the top of XML pages
and are utilized
when XML code
is compiled into its
final output format.*

is an international company dealing with a host of different countries, Web pages must be produced in a myriad of different languages and styles. XML allows developers to create a central repository of information and then make changes to the presentation of that information based on their customers’ different needs. Also, since some elements are the same regardless of their geographic point of reference—logos and graphics, for example—these can be contained in one area and then duplicated elsewhere on the site.

*The majority of
XML coding, including
the actual XML,
the XML schemas,
and style sheets,
are invisible to the
end user.*

XML can Adapt with the Times

The ability of XML to be ported to multiple outputs is also important. HTML is the most widely accepted language today for Web use, but this may not always be the case. Because XML allows for data to be ported to different types of outputs, data can later be adapted to changing industry norms. Whether or not HTML stays in vogue is unimportant—XML has the ability to change with the industry.

The Transition to XML

Dell’s transition to XML was a process that involved not only moving existing content from HTML to XML, but it also involved ensuring that the content being transitioned was relevant to Dell customers and the focus of the site.

The first step in the process was to examine all of the existing content and, without regard for personal interest in the content we had worked so hard to develop, decide what content was important for the new site. Re-creating the site as efficiently as possible became the new goal. We looked at a site that had brought us considerable success in the past and tried to decide how we could take a winning concept even further.

This transition in the site required us to make many changes to the way we used the data. Gone were the days of static content that had to be replicated over and over across the site. XML made content—from what used to be completely separate areas—available throughout the entire site. And since the site is global, it was important that it provide the same customer experience to all customers, regardless of their location.

XML Easily Streamlines the Dell Home Page

The Dell home page (www.dell.com) provides a great example of this new innovation. Once a cluttered page

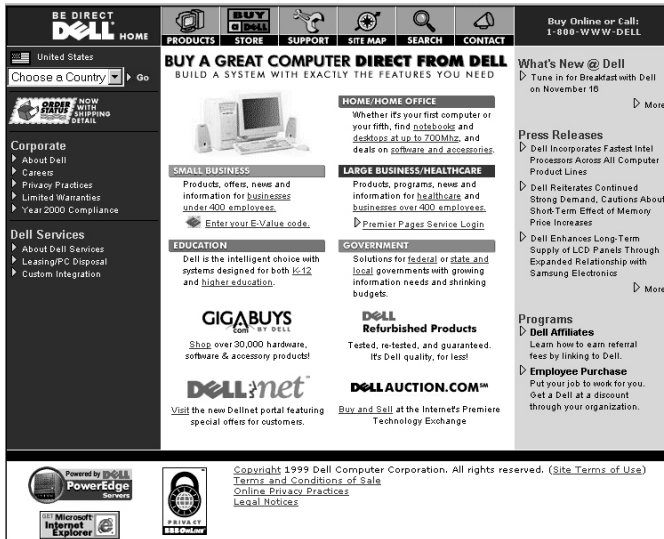


Figure 1. The Old Dell Home Page

with superfluous links to varied content throughout the site, the new front page is clean and clear. It contains far fewer links that bring customers immediately to their business segment-specific information.

Figure 1 shows the previous Dell home page created using HTML. Figure 2 shows the new XML version.

When we transitioned the site to XML, no XML infrastructure was in place, so we were able to re-create our development process from the ground up. Although XML is a different language than HTML, many coding details are similar. These similarities allowed our developers to use some of the familiar tools and processes that they had relied upon in their development of HTML.

For example, many developers used applications such as Allaire HomeSite™ and even Microsoft Notepad for their HTML development. Both applications can be used in the new XML environment, and HomeSite 4.0 even offers the ability to add XML-specific “snippets” to a special reference area on the application. These “snippets” can be easily used during XML development. This interoperability of the applications helped developers adapt their current tools and technology to better fit in the new environment. In the end, all of the hard work of Dell’s developers has created a new site far superior to the old Dell site.

Because XML allows for data to be ported to different types of outputs, data can later be adapted to changing industry norms.

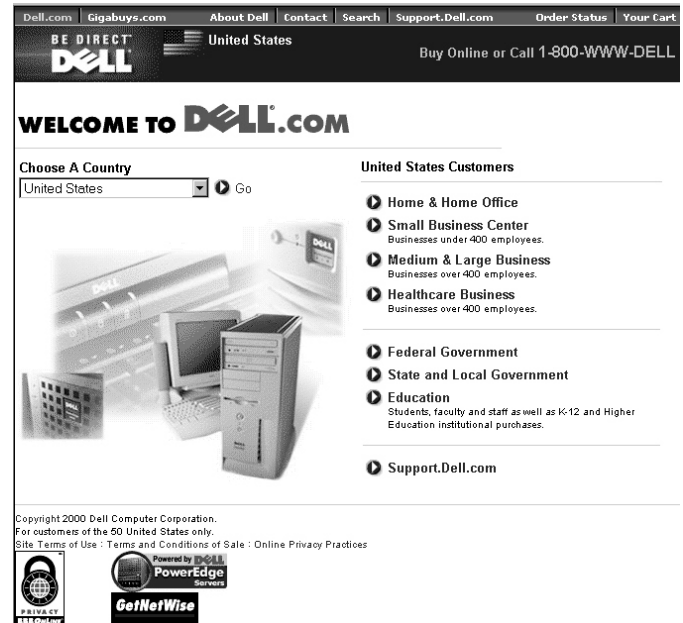


Figure 2. The New Dell Home Page

XML and the Future

Initially, one of the biggest reasons for a push from HTML to XML was the desire to create a language that could be read across a large variety of machines and many different users. Unlike HTML, where data is fixed and different style sheets can change format, XML has the ability to provide the same data to a myriad of different sources, with the option of many different outputs. XML, if subjected to guidelines and constraints that everyone agrees to follow, could create a new language that allows machines to communicate among each other with greater efficiency than ever before imagined. ♦

Paul Laster (paul_laster@dell.com) is the Web developer for Dell Enterprise Systems Group (ESG) Global Alliances. He has been with the Alliances team for over two years, maintaining both its internal and external Web sites. He is also the primary technical contact for the Dell Direct Effect Alliance Program. Paul has a B.S. in English and Business from the University of Texas at Austin.

**Test your applications before they go live.
Visit a Dell Application Solution Center (ASC).
For more information, see ASCs at www.dell.com.**