

Benefiting from the use of XML does not require complete replacement of an existing Web site environment (such as .NET, PHP, JSP, Java, ColdFusion, Dreamweaver, etc.), nor even altering it significantly beyond the addition of some XML. It is more important to determine **where** to incorporate XML within the Web site to make the greatest impact.

Implementing XML on part of a Web site, rather than the entire Web site, often proves to be the most advisable first step. It makes more sense to focus on those areas of a Web site that are not managed efficiently and seem to be appropriate for an XML approach, such as those containing:

- publications or large amounts of text organized into pages and linked;
- repetitive content or similar formatting;
- identical information in different formats (such as HTML, PDF, printer friendly) or for different devices (computers, PDAs, cellphones).

In this approach, parts of the Web site may be developed as XML, while the other parts remain unchanged. This “mixed” arrangement may be the best solution in some cases, especially where the non-XML-based areas are already highly efficient and would not really gain much value from a conversion to XML.

However, in other cases, the benefits derived from the partial implementation lead to expanded uses of XML in additional areas of the Web site. The efficiencies derived from using XML in one area creates opportunities to apply those efficiencies in other, less obvious areas. Ultimately, the entire Web site may be converted to XML. (This is exactly what happened in CTG’s experience with XML, starting with its publications area and moving out to the entire Web site.)

This complete transformation of a Web site to XML means that all the content originates from an XML format, either as static XML files or as dynamic data retrieved from a database and reformatted as XML. Likewise, all the HTML pages (and potentially PDF and other pages) that comprise the Web site are produced automatically from XSL files.

In other words, the individual HTML files are not created by Web developers and saved on a Web server (or generated dynamically via script files as in database-driven sites), then retrieved as visitors request them via their browser – which is what happens in a typical Web environment. Instead, an XSL file selects content from a designated XML file (or files), transforms it into the appropriate format and appearance, and delivers an HTML page to the Web visitor’s browser (see Figure 4; click on figure to display a larger image).

Figure 4. Creating and Maintaining HTML Web Pages via XML/XSL Files.

The difference is that one XSL file can produce multiple HTML pages, and one XML file can also be transformed into multiple HTML pages. Hundreds or thousands of individual HTML files are replaced by dozens of XML and XSL files, which reduces the maintenance effort and enhances consistency.

However, for both partial and complete implementations, certain technical steps need to be taken because no currently existing Web environment is completely “XML ready.” Most Web servers, for example, are not currently configured to automatically process XML files and produce the desired output (HTML pages) for the user. Some browsers, on the other hand, can process XML and produce HTML, but this processing ability is limited and unpredictable.

Fortunately, Web servers can be configured to support XML. There are three basic approaches:

- Implementing an XML framework, which is software designed to handle the XML/XSL processing for a Web server. Apache Cocoon is an example of a Web development framework. (CTG uses Cocoon for its XML-based Web site: <http://www.ctg.albany.edu>.)
- Employing configuration files and server-side scripts to dynamically intercept and transform Web requests and responses (HTTP). These are relatively simple scripts that tell a Web server how to find and process XML/XSL files. Different scripts (written in ASP, PHP, JSP, C#, ColdFusion, or Java, for example) will be suited to different Web environments
- You may also incorporate XML via “include” files and other insert capabilities native to Web design environments such as Dreamweaver or ColdFusion. In this case, you would not be changing your current Web practices at all, but just enhancing its capabilities.

Implementing XML in Your Web Site Environment

Note: *CTG's XML Toolkit Web Site, <http://www.thexmltoolkit.org> is a good source for coding samples and tips to assist developers learning how to use XML in Web sites. The code samples are provided for a variety of Web environments in a modest, moderate, and elaborate framework with clearly explained learning steps that address specific topics (see Figure 6 on page 15).*