

Why would a business use XML?

What this chapter covers:

- The benefits of XML to a business
- Design considerations
- Scenarios

3.1 *Overview*

Let's imagine the company Resumes R Us! is a one-person Internet startup. It intends to gather and distribute resumes from software developers and distribute the information across the Internet. Its competitive edge will be the customizable and immediate way it can provide its data.

3.2 *The business problem*

The clients require the resumé data to be available in different formats. Some clients dial up via modem and cannot take images; other clients require completely formatted information, which they can simply print off. Still other clients simply want the information to be sent in a recordset form that they can incorporate into their databases on a regular basis.

Some users cannot be limited to using only programs written in Java, Visual Basic, HTML, or other languages; they must be able to query and receive replies across mediums such as email, HTML, or perhaps even a telephone. Still more clients will not be making the requests for information themselves, but will reply upon automated programs to fetch the data at irregular intervals, requiring immediate delivery of the information in the requested format.

For Resumes R Us!, this wide variety of requirements poses substantial challenges, both in technology and in time to provide these services. Since this is currently a one-person startup, these services need to be automated and flexible. Ideally the solution provided will enable the client to design their own solutions and relieve the small company from the workload. Also, if Resumes R Us! adds more information as it becomes available, the computer programs that the customer uses must not be made unstable at all.

How can XML help us achieve these grand goals?

3.3 *User scenarios*

Here is a selection of scenarios where the customer's needs are met by our XML-oriented designs.

3.3.1 *Scenario 1—Bandwidths and customization*

The client requests that specific data be returned in a specific format. A selection is made in a Visual Basic application or on a web page, and the submit button is pressed. The reply must be received quickly and with the requested format.

The business problem for Resumes R Us! is that the format could be in a huge variety of colors, sizes, and bandwidths! The client may want summarized financials, tables,

images, colors, or simple text for printing immediately. Clients may also want the data formatted for high-bandwidth online viewing with no database access (images included, maximum information) or low-bandwidth online viewing with no database access (no images, less information).

Behind the scenes, the standard XML is transformed using XSL *on the fly*. What this means is that one dataset can be returned in many different formats. Now Resumes R Us! can associate different bandwidths and customization requirements with each of their clients and apply the appropriate XSL stylesheet in order to fulfill each client's needs.

3.3.2 Scenario 2—Immediately usable data that is reliable

The client has set up a program on its server that will automatically request data at intervals. The data should be returned in a package that the receiving program can unpack and use immediately. Robustness is a vital issue for this scenario.

The Resumes R Us! business problem is that the contents of the database will change from time to time, and it is important that the solution that is eventually created can handle all the changes. Not only that, but the customer's software must also be able to extract the fields it is expecting without crashing when it comes across data it didn't expect.

Behind the scenes the data can be returned in the format of:

- XML text file with a DTD (either inline or referenced externally), and/or
- An ADO 2.1 persisted recordset file format.

The Microsoft ADOs are recordsets with functionality similar to RDO and DAO recordset functionality.

When using an XML file, the receiving program can read the DTD and look in the XML file for the data it needs. This removes the issue of programs expecting certain data formats, exact placing of characters, or consistent columns in comma-delimited formats. The data is also immediately usable. If the solution uses the ADO 2.1 persisted-file approach, the file is reloaded from XML back into an ADO 2.1 recordset and is immediately available for use as a data-source object.

3.3.3 Scenario 3—New layouts

The customer regularly has a new layout for their data and must receive the information from Resumes R Us! in the new format promptly. Any delay in the turnaround would have a negative impact on the relationship with the client.

The business problem is that Resumes R Us! cannot afford the time to extend the existing Visual Basic programs and cannot afford to be seen as ignoring the custom needs of the client.

Behind the scenes, the client designs their own layout using XSL and uses a web page or small Visual Basic program (using FTP) to upload the new XSL file to the server. When Resumes R Us! receives the new XSL file, it is immediately available in a layout in which the client's data can be returned.

Note The ability for the ADO objects to save the recordsets as XML is provided by version 2.1 of the ADO objects. As long as the file is not changed, another program using ADO 2.1 can unpack the recordset file back in the original XML text file.

3.3.4 Scenario 4—New requirements

Occasionally Resumes R Us! creates new ways to retrieve resumes from their database. One idea they are toying with is to email their customers a contact telephone number that can be used to listen to the latest resumes. This service also guarantees that the voice messages are up-to-date and not prerecorded.

The business problem for Resumes R Us! is the time and cost involved in setting up these new solutions. This solution in particular cannot be a prerecorded voice; it must be dynamically generated on the fly in order to be up-to-date.

Behind the scenes, using XSL, Resumes R Us! can return data to the clients in almost any format that an XML-aware text-reading program can read. A suitable application for reading the XML file would be an application that manages call centers.

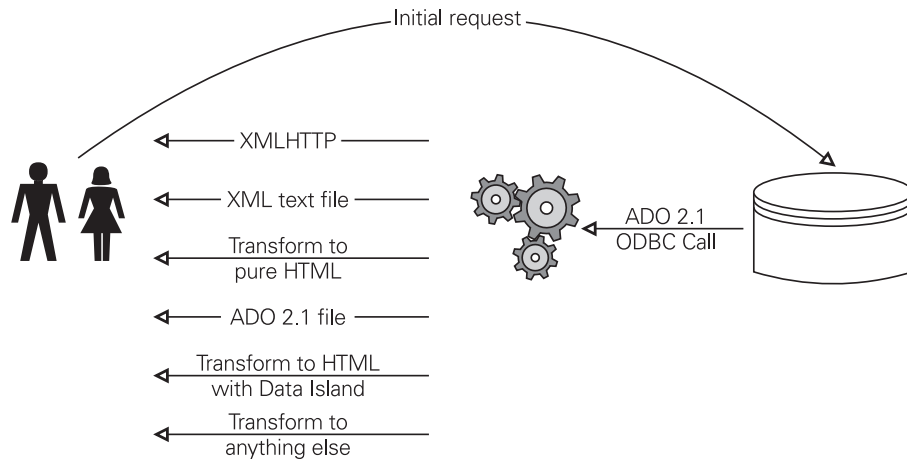


Figure 3.1 The XML possibilities are endless

3.4 Solving the problem with XML and XSL

One of the key patterns we see above is that XML combined with XSL can provide the same data in several different ways. This is a key benefit of XML; as a text file, it can be read, written, and passed across all platforms, as shown in figure 3.1.

It's beginning to sound like XML is the all-healing, world-peace-inducing snake oil for computing! Sadly, it is not, but it does bring some useful characteristics to the computing universe. Read on to find out how to use it in your applications and solutions.