

Designing a dynamic XML website

Dan Riggs

OpenMIND Consulting, Bicester, United Kingdom

driggs@open-mind.co.uk

<http://www.open-mind.co.uk>

Abstract:

This presentation is a case study about the implementation of a 100% XML, dynamic website for a top European automotive manufacturer.

The aim of this presentation is to provide an overview of the decisions involved in planning and implementing such a website.

The initial focus will be the analysis of the original problem faced by the client, which created the need for a new system. An investigation revealed several issues the client wished to resolve. In practice all of them could not be addressed due to time and budget constraints, therefore it was quickly established which of these issues had priority. This allowed the final project requirements to be defined.

The use of XML for this project was not a forgone conclusion at the outset. However, once the project requirements were understood, the decision to use XML became clear. This presentation will examine the choice of architecture for our solution, and the decisions leading to its selection.

Many factors were considered during the design phase, including the target audience and their available technology, as well as issues like download time and the number of concurrent connections to be handled. These and similar considerations encountered during the project will be covered.

With the many XML related products available, an important part of the project was assessing which tools to use. The choice is largely dependent on the approach chosen. The selection criteria used in this solution will be covered.

A look at inner workings of the solution will provide more detail about the decisions behind the DTD design, as well as the coding and design of the website itself. The website design included considerations such as the appearance of the site, the site navigation, and how to handle hyper-linking between the document fragments - all important issues to address, and all issues which could have been approached differently depending on the requirements.

The XML source for the website required creation and existed initially as Word files. Authors unfamiliar with markup usually take time adjusting to using an XML editor. The presentation will cover the decision to re-author the content, and detail how an XML editor was configured and an authoring team trained to allow production of the XML source.

To conclude, there will be a brief summary allowing delegates to decide how successful the project was in terms of meeting the original goals set.

The presentation will include a demonstration of the solution discussed.

Introduction

Requirements

To set the scene, our client is a well known car company. We had set out to assist with the translation of the client's dealership handbook. This required translation from the client's native language to English. It soon became apparent that our client had more problems than just translation.

The main problem was one of distribution. The paper handbook had to be delivered to hundreds of dealerships around the U.K. The handbook was continually subject to revisions, and the client had problems in ensuring each dealer had an up-to-date copy. Having received a revision to the handbook, it was left to the dealer to ensure that the update was filed correctly.

A secondary problem, was that each of the manufacturer's marques required a unique look and feel. The actual content was also subtly different between marques, so a customized version had to be produced for each one.

A system was needed which could handle:

- Frequent Updates - as much as 40% of the total content could change in the space of a month

- Consistency - allowing all dealers access to the same, up to date information

- Dynamic content - Reprinting and content restructuring dependant on marque

- Security - preventing casual users from seeing more sensitive information

How difficult could it be?

Using XML

How

We considered several potential ways of meeting the clients needs:

- ASP/PHP or similar, serving dynamic pages of HTML

- Database serving HTML

- Static HTML

- XML based client-server solution

Our choice was to design a website, which used the power and versatility of XML to deliver the handbook to the dealers.

It is interesting to note that the first two technologies listed above, can actually be complementary to XML.

Why XML?

Good reasons to use XML:

Standardization - XML is a W3C standard therefore the data format should be futureproof

Readability - XML can be equally readable to both human and machine

Validity - Parsing XML against a DTD ensures consistency of document structure, preventing authors from imposing their own style on the document

Independence - Platform and software independent, even if we change our software, the data does not need to alter

Versatility - Easy to transform XML data, so you can convert your data from one schema to another. Allows output to multiple formats HTML, PDF etc.

Reasons not to use XML:

Knowledge/skills - or lack of it where XML is concerned; 'great technology, but how do we use it?'

Trust - Lack of trust in emerging technologies, the wait and see approach

Cost - Investment in new software and training

Time - Authoring time can be increased slightly as you need to add all those funny tags...

So there is a downside to using XML then?

The truth is that XML is simple to learn from a programming point of view, and anyone familiar with a word processor will not take long adjusting to XML authoring. This means that the skill gap can be bridged quickly and easily.

XML is here and people are using it. People who are adopting a wait and see approach should be at the point where they have waited and are now seeing the results achievable from technology that uses XML.

Cost is always a problem, but if you are investing in a website, you will need to spend money on software, development and training regardless of the technology used. An XML website need not cost any more than a well designed non-XML website. If you have a website implemented 'in-house' you are probably halfway there already in terms of software and skills.

Developing an XML website may take slightly longer in terms of content creation and possibly even development. But this is the price of reuse and dynamically changed and presented content.

How much is XML going to cost?

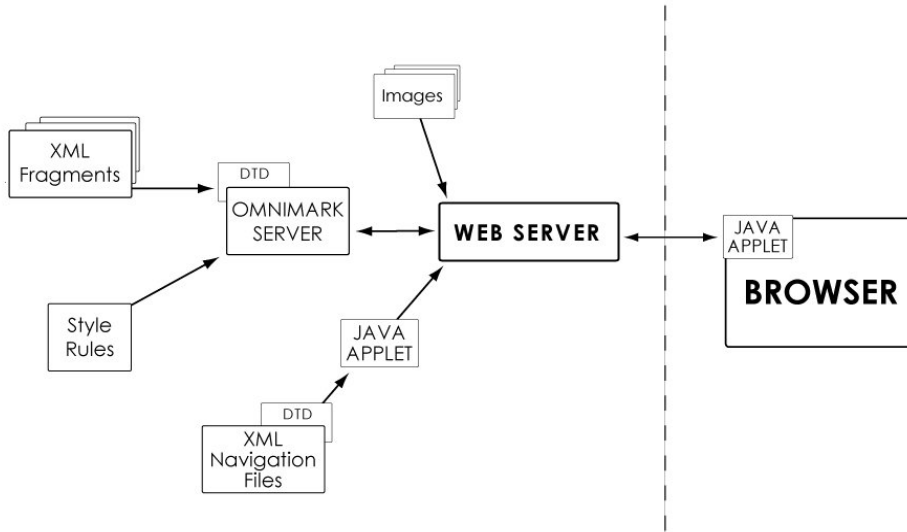
An important choice to make, is whether to pursue a custom solution or buy one 'off the shelf'. An 'off the shelf' solution can potentially save you time as a lot of the work has been done for you, and the initial cost can be relatively high. A custom solution is probably cheaper in the short term and good as a 'proof of concept'. It is likely to lack the functionality of the 'off the shelf' products initially, and will take longer to develop for a fully fledged system. You do however get the benefit of a solution which is adaptable to your exact needs.

The initial solution developed for our clients took approximately nine months from concept to implementation, with costs comparing favorably with publicly accessible e-commerce sites. At the clients request we are extending the functionality and continuing to develop the solution.

Whatever architecture you choose, compared with the cost of producing the equivalent in paper manuals, an XML solution does not take long to pay for itself.

Dynamic website design

XML makes the concept of dynamic documents a real possibility. By manipulating XML fragments, information can be repackaged to suit individual needs.



The ability to separate document content from presentation (see diagram) was vital when it came to producing a handbook tailored to each brand. Font, colors, logos and even the interface design can all be separated from the content. The style information can then be added automatically when the document is requested. Keeping branding and style information separate, you can change the style in one place and the change is reflected everywhere.

Our client also desired the ability to alter the document depending on who was reading it. This allows several problems to be addressed:

- Security - Potentially sensitive items can be hidden from the view of casual users

- Applicability - For experts familiar with the documentation, only pertinent information is shown, keeping the padding hidden

A modular document design allows us to tackle these problems by delivering the appropriate mix of XML fragments about a subject, configured to match the user requirements.

Granularity is an important concern in a modular document design. It represents the level at which you decide to 'chunk' your data into manageable and reusable segments. The trade off is between reuse, and configuration to the nth degree and the complexity of design required to make it happen. For this solution we decided to set the level of granularity to the 'topic' level. That is to say that approximately one page of information would represent one complete XML fragment. To set a lower level would have allowed more accuracy and control over the content, but would have increased implementation time.

Another important decision to make was client Vs server processing. In other words do we want to have our server doing all the work, or rely on the user having the appropriate browser technology to take some of the processing load. We initially opted for 100% server-side processing, but have since moved to client-side navigation processing. This increases navigation speed on the client machine, but results in slightly increased initial download time. It is important to consider the level of browser technology available to your users. By not having 100% server-side processing we are potentially sacrificing compatibility with older browser software and tighter security controls in exchange for speed and reduced reliance on the server.

Authoring the content

Once your implementation is ready, someone will have to write content for it. This is not as easy as it sounds as the authors must understand how all the elements in your DTD are to be used. Without some degree of training or at least some usage notes, authors will author but not necessarily exactly the way you intended them to. If your DTD contains element names which are ambiguous, authors will misuse them and create content which may cause problems when displayed or referenced, and may not even work at all.

The DTD used for this project was one created specially for the client. The structure was organized so that large portions of the document could be authored at once, but with the intention of 'chunking' the 'topic' elements into separate, unique XML fragments at a later stage in the content creation process. The reason for not authoring the topics separately from the start was that the authors were working from existing paper documentation. The client wished to retain the existing document structure, and in order to minimize complexity, it seemed sensible to author at a higher level than the resultant fragments. Now that the bulk of the content is authored, 'topic' level authoring is desirable as it allows multiple authors to 'check out' and amend individual document fragments.

No problems?

In many IT projects, the biggest problem is populating the data source. This project was no exception. Probably 25% of the time was spent creating and implementing the solution, and 75% of the time was producing the content. The handbook was stored on paper and as Word documents. Scanning, converting and proofing the content would have taken too long for our timescale, and no conversion software was available to us. As a result, we actually re-authored into XML by hand (using a good XML editor!). If you are planning a website with a large volume of data, it may be worth thinking about a data conversion program to speed up content creation.

XML authoring is fairly simple to learn. That said, for this project we attempted to bring in some external support for our authoring team. Due to the time constraints of the project, however, we could not afford the time required to train all of the extra authors to use the XML authoring tools. This caused us some problems in getting the content converted to XML.

Due to the need to get the solution up and running quickly, the granularity was set at quite a high level. The fragments were quite large which makes for a less elegant and re-usable solution than we might have hoped for. On the positive side our solution is not too complex to use or maintain, and we plan to increase the granularity as development continues.

As the solution developed, the requirements changed slightly as we could see the potential for delivering more and more functions. Scope creep is obviously a common problem with custom built solutions, and it did not help that we changed the technology behind our solution several times during development. This problem was mitigated by the fact that we had envisaged an iterative design and implementation process.

Design changes were fairly simple to make with our technology selection, and we can continue to extend and improve the solution over time. It is a testimony to the versatile nature of XML that we were able to make considerable changes to our solution with minimal affect to the content.

What next?

Our original challenge was to match the quality of the existing paper-based documentation. Having achieved that, our challenge is to make the solution all it can be.

Something that we are currently implementing is the idea of multiple document 'views'. In other words, allowing different ways of accessing the information, for example:

- Linking content to images allowing graphical navigation

- Multiple TOCs with a bias to supporting a particular topic or process

- Dynamic topic groupings through X-refs

- We want to be able to add new views as desired.

Developments to look forward to include:

- A drag and drop document structure editor to allow the creation of new 'views' which will be stored as XML documents themselves

- Online authoring to support authors across different geographic regions

- Adding more language support to the solution (we already support several different English versions of the content)

- Storing graphics using XML rather than the formats traditionally used on the web. This allows graphics to be searchable and editable by the user.

Conclusion

XML is now firmly on the map, and the number of XML web solutions is steadily increasing. Hopefully, it will not be long before everyone can enjoy the benefits of using XML, and dynamic XML websites will become as common as static HTML websites are today.

Author

Dan Riggs

Consultant

OpenMIND Consulting

Postal Address:

Cherwell Innovation Centre, 77 Heyford Park,

OX6 3HD Bicester

Oxfordshire,

United Kingdom

Telephone: 01869 238080

Fax: 01869 238081

E-mail: driggs@open-mind.co.uk

Web: www.open-mind.co.uk

Daniel Riggs - Dan Riggs is a Consultant with OpenMIND Consulting, and has been involved since 1998 in developing XML solutions for several major automotive clients.