**E-Commerce website design: Keeping It Ximple**

Michel Jouvernaux  
Waikato Institute of Technology, New Zealand  
Michel.Jouvernaux@wintec.ac.nz


**Abstract**

XML was carved out of SGML as a basic building block of the 'Semantic Web' of tomorrow. But before this 'brave new Web' can become a reality, there needs to be a radical change in the techniques underlying web site design. While some debate whether to allow XML in their databases, this paper demonstrates that for the small to medium size e-Commerce venture, XML may well be the future: by exchanging the common N-tier architecture used today in favour of a simpler 2 tier with XML and XSL. After all, XML documents are data sources. For the smaller organisations most commonly found in New Zealand, the pursuit of competitive advantage requires a careful targeting and tight control of resources. The structure of retail e-Commerce data makes possible the use of building techniques using the XML/XSL combination. Pivotal to this concept is the separation of contents and presentation made possible with XML/XSL and the innate advantages of a '2-tier' architecture, with simplicity, cost, reliability, and the efficiency gains of distributed processing. An example prototype web site demonstrates these techniques and the technology requirements.

**Keywords**

XML, XSL, semantic Web, e-Commerce

**1. The Small e-Business and Competitive dvantage**

Throughout the 1980's and 90's, Information Technology (IT) was seen by many organisations as the key to gaining a competitive advantage in their respective markets (Gates, 1999; Porter, 1985). Around the turn of the century, many companies spent valuable resources creating a presence on the Internet but too often this was done merely as a response to their competitors' moves in cyberspace; many such forays in "e-Commerce" were poorly planned and very few companies actually gained a decisive advantage from it. The "dot com bubble" and its subsequent burst showed that resources were often squandered on e-Commerce projects that had no real chance of ever showing any benefits. A simple principle had been overlooked: Competitive Advantage is not gained because a particular system is beautiful, or expensive, or cutting-edge, or built with fashionable technology, but simply because it is more efficient, either in operational or financial terms, or both. A system is efficient if the cost of building it is considerably less than the revenue that can be expected from it (sometimes popularly expressed as "value for money"). The rate of return is normally calculated as part of a project's risk analysis, but often not looked at anymore once a system is in place and hard data can be obtained (cost of operating new system vs. cost of operating old). This "financial" efficiency should extend to each task performed in that system; it should not cost more than its potential benefit, and ideally perform better than competitors' systems. This must be balanced with the "operational" efficiency, which is related to meeting the requirements and expectations of the business and its clientele. Some of these expectations are obvious: the system must "do the job", i.e. deliver on its basic promise. Others are more subtle: the end user
might put up with some quirkiness or lesser performance in exchange for a real or perceived gain (e.g. security, privacy, reliability, availability).

Most New Zealand businesses are small to medium size companies. This doesn't mean that Porter's rules do not apply to them; but they have comparatively less resources to expand trying to gain a competitive advantage. Taking the e-Commerce example, many systems are actually over-engineered in some areas (so not "financially efficient" as more resources are spent than was necessary) and still not meeting end-users expectations or basic requirements (so not "operationally efficient" either). One major block to true efficiency of e-Commerce web sites is that smaller companies often have to outsource the hosting of their e-Commerce web sites. This means they are limited as to the technologies made available to them by the hosting company, generally an Internet Service Provider (ISP). The typical architecture employed is N-tier, where data is stored in a 'back-end' database server; the presentation is achieved by some processing logic creating the necessary HTML code and passing it to the web server, which in turns sends it to the client computer who requested it. Undeniably, this architecture does achieve a dynamic "data-driven" presentation of the commercial offering, but such complexity is often unnecessary. It also means that in most cases the business loses control of its data and relies on the ISP for its update. This is costly and inefficient, and, what is probably worse, it effectively locks the business with that technology and thus that hosting company, re-enforcing the power of the supplier in the business relationship (Porter,1985).

System designers nearly always apply designs and technologies that they are familiar with, because of the personal and commercial investment that they have in those, regardless of what is the best (most efficient) solution to the problem. This means that the efficiency of building the system - a worthwhile goal in itself - takes precedence over the efficiency of its operation. The business has to live with the consequences for the life of that system. In fact, simpler, cheaper solutions are thus often overlooked, among which the rich set of technologies and tools available in the public arena. Open standard technologies, essentially free for all, can be used to create efficient and effective e-Commerce sites designs. Among those XML stands tall, and also holds the promise of tomorrow's World Wide Web. This paper attempts to demonstrate that a simpler Web site design using XML can meet the quite complex and dynamic requirements of a retail e-Commerce web site, and also be a step forward to the W3C's goal of the Semantic Web.

2. XML And The Semantic Web

There is no need to introduce the eXtensible Markup Language (XML) here. It is sufficient to remind everyone that, like HTML, it is a subset of SGML, and the World Wide Web Consortium (W3C, 2001) is the body that controls both HTML and XML recommendation. Of importance to our discussion is to introduce (or remind) readers of the long term vision of the W3C and its famous director Tim Berners-Lee. That vision for the near future is called "The Semantic Web".

"The Semantic Web is an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation."


In a nutshell, the Semantic Web requires the addition of metadata to web contents, enabling the use of software technologies like Intelligent Agents to grasp and process the meaning of the contents. This is a major advance on the "old" web where documents are mostly in HTML format, which is concerned only with presentation. In the Semantic Web, separation of contents and presentation is achieved, an important aspect of Semantic Web Design (Goldfarb & Prescod, 2002)

XML is but one of the set of technologies required in achieving the Semantic Web vision. RDF, Ontologies and Agent technologies are equally important. But logically following from the above arguments, for the Semantic Web to work at all, web sites contents will have to be based on XML documents. A current trend is to create XML as required from
data, or even store XML itself in a database. In many ways this kind of architecture will not deliver the Semantic Web of tomorrow. Agent based components of the Semantic Web architecture will in most cases not be able to access the database-stored data and so may get an incomplete picture of what a page is about. Furthermore, the complex technology of a backend database and the processing required to transform raw data into "browser friendly" documents have a profound impact on cost, reliability and performance. And often it simply is not necessary, as the XML set of technologies provide all the tools needed to achieve "data-driven" web sites.

XML is designed for the easy and reliable transmission of data across computer systems (Harold, 2004). This is exactly what a web server/web client pair does. In an e-Commerce context, making reliable product information available to potential customers is the end goal. To achieve that goal, XML technologies also offer all necessary features. XML is easily validated against a DTD or schema, and easily transformed into another XML document or even another format using the W3C's eXtensible Mark-up Language for Transformation (XSLT). XML is easily transmitted over the Internet, and need nothing more complicated than a text processor to be read at its destination. It is very simple to generate web site contents in a ready to process and cross-platform format like XML. A presentation layer can be added and tailored to the capabilities and/or requirements of the client:

- An Agent will likely require none, or may choose from the different style sheets available which suits it best.
- A browser or other software can be satisfied by using either server processing technology (SUN-Apache Xalan, Apache Cocoon) or even better take advantage of the (distributed) processing available on the client itself and use XSL transformation style sheets (XSLT).
- A mobile device (cell phone, PDA) can be served the exact same contents but with a different style sheet meeting the requirements of mobile standards (e.g. WAP).

The Web Browser of the future will only need to understand XML and XSL, and resulting HTML will be rendered on the fly (Moller, A, & Shwartzbach, M., 2003). What is not widely realized by developers is that such web browsers are already common place: Microsoft Internet Explorer 6.0 and the recent Mozilla family of browsers (Netscape version 7.0, Firefox 1.0, Mozilla 1.7) all include an embedded XSLT processor readily accessible to the web designer. What is more, interaction is possible between well known core web development technologies such as Javascript and XSL, opening possibilities like the parameterisation of style sheets based on cookie values or system variables. XSL is in fact an amazingly versatile and capable programming language.

It is the opinion expressed here that the Semantic Web will require a web site's contents to be created entirely in XML and stored as such on a Web server (where any Intelligent Agent will be able to access it). A web site operating along those principles will be in step with the evolution of the HTML based Web of today into the XML based Semantic Web of tomorrow.

3. A Hierarchy of Data

A lot has been written about the capability or otherwise of XML as a database; certainly, comparing XML to a full fledged RDBMS, the shortcomings become glaringly obvious (Bourret, 2003), even if XML is undoubtedly a data storage of some kind. XML is not going to replace database management systems, but XML is great to hold data for transfer between databases or from a server to a client (Harold 2004). Naturally, if we want to present data on the web, that's very much the case.

Our goal is a web site where changes in the underlying data - the contents - will automatically be reflected in what is presented to the viewer. So to create a web site from an XML data source, the "shape" of the data has first to be considered. Because XML documents are essentially 'tree-like' in structure, only data that lends itself to this kind of structure will be able to be represented by XML; if the "shape" of the data is right, then a RDBMS - and all the associated complexity - is simply not needed. Depending on its source and usage, data can be highly relational and normalised, or alternatively of a hierarchical nature. An e-Commerce retailer's range of products is
nearly always arranged in a 'tree' structure: e.g. categories, sub-categories and finally
detailed product information. This kind of data lends itself well to be stored in XML. Of
course, one agrees that careful data analysis is needed for to the optimum tag
arrangement for our XML database, according to best practice and modelling ideas
(Kennedy, 2003). But what makes XML great is that changes to XML structures can be
done quickly and cheaply, and the techniques used to read XML - parsing - means that a
structural change does not automatically require changes in the programming logic of
the presentation layer (this is nearly always true as long as tag names do not change).

Any file transferred to the client browser should be an optimum size (ideally no more
than 60k, or a 10 second download) (Stockley, 2004). Data can be arranged in multiple
XML documents for the needs of the system, broken according to the product hierarchy.
There are no limitations in the practical range of products that can be displayed this
way, but admittedly it will complicate document management. This problem is tempered
by the observation that most e-Commerce sites have a limited range of products on
offer. Admittedly, if products do change often or count in the thousands, the relational
database approach is probably warranted. Alternatively, extracting "refreshed" XML
product files from the database, rather than use the database itself for the site is in
most case neither difficult nor expensive.

This hierarchy is also useful as the navigational path to the product. It facilitates
presenting other products in the virtual store along the way. The design should be
balanced so that the user can get to what he wants quickly enough, without
frustrations, but still making good use of the marketing opportunity. The user will not
likely know everything that the site is selling - so "search" in the conventional web
sense - 'a la Google', on string match - can be detrimental to sales in these situations;
leading the user in a straight path to one and only one product is wasting the marketing
opportunity.

4. The Battle of the Tiers

The most common design for e-Commerce web-sites today is based on N-tier
architecture. This consists as a minimum of a client, a web server, some application
logic and a database. Web pages are created dynamically from data stored in the
database, in response to preset queries or user requests. This architecture has many
advantages like easier management of site contents, different levels of security applied
to different contents and the possibility of full text search on all the contents (although
as previously said that may be of doubtful usefulness to the e-Retailer). There are also
many inconveniences to the design:

- The necessity of communicating with the database in the first place means an
extra sizeable overhead is placed on the response time to an HTTP request
- Maintenance/Synchronisation of the site data with a production database, which
often necessitates downtime to be accomplished.
- Scalability: the requirements for hardware and/or network bandwidth between the
web server and the database increase relative to the number of requests to the
site. This can be very hard to get right, and if gotten wrong costly in terms of
missed sales. As more people connect to a site, the overheads to the server pair
(web and database) increase exponentially. The database is often overwhelmed, as
its work is much more complex than the simple serving of a text document (the
web server's task). This results in much delays and frustrated web surfers.
- Reliability is a major consideration, and the more components to a system the
greater the risks of failure. If the data source fails, or the application layer fails,
the whole system fails. And it happens!
- The cost of the extra hardware and software licensing (RDBMS, middleware, etc...)
is also an added (often yearly!) burden.
- The cost of the extra technical maintenance required can become an obstacle to
the small, even medium-sized web entrepreneur.
- The loss of control over the development technology. Smaller firms have to
develop a site based on whatever RDBMS and development technology their
access supplier (ISP) supports. They often cannot move it easily to another host,
and have little control on the updating of contents. They cannot move the site
easily to another service provider is a disagreement occurs.
The argument presented here is that the cost of hardware and the complexity of the 3 tier model are surely not necessary for every web site. As explained in the previous section, a set of XML documents can be a database, and thus a data source to a web site; but these documents are merely text files and are easily and effortlessly transferred to a client from any web server. The processing required for the presentation of the XML contents is done by the client web browser, using a style sheet, and this includes sorting query operations. The XML/XSL combination allows a data driven web site at a fraction of the cost (no licensing required, ISP needs not be involved), delivering great performance (not need to connect to the DB server every time) and reliability (less components). What we have is not only a simpler 2 tier architecture, but also a great example of distributed processing.

5. An XML Based e-Commerce Web Site

Our example is based on a real-life company and its e-Commerce requirements. Le Gastronome Ltd specialises in the sale of French delicacy products. They sell several range of products across 3 separate channels: wholesalers (supermarkets), specialty shops (delicatessen) and direct to the public. They want a web site that can help present their products to their buyers, and also accept orders and payments online. Users can log on to the site and will view different products and most importantly different prices depending on their channel status. A prototype XML based site was created; below is a run down of its components. For more details, refer to the accompanying site map (Figure 1).

5.1. The Server

The server could be any "run of the mill" Apache. Due to the nature of the site, merely transferring text files, any web server software would do, making the site highly portable. PERL was chosen for CGI requirements so as not to compromise this portability (as it is standard on most if not all web servers).

5.2 The Client Browser

Browsers such as Internet Explorer 6.0, Netscape 7.0 and Mozilla 1.6 support XML and XSLT reasonably well. The site being after all a prototype it is felt good enough that it worked 100% with Internet Explorer version 6.0 (IE 6.0). That browser has 68% of the browser market to date (Onestat, 2004), with all versions of IE accounting for nearly 95% of browser use on the Internet. Note that known compatibility issues between the site and Netscape or Mozilla hinge on the scripting used (JavaScript), not XML and XSLT transformation. These will need to be addressed for a commercial version.

5.3 User Login

For the security necessary around the login process, user information is contained in a secure XML document on the server (in fact outside of CGI-scripts folder and outside the root of the web server, thus inaccessible to clients). The process uses PERL and the CGI standard. The XML document can be efficiently 'slurped' into a PERL hash structure (associative array). On validation of the user credentials, a client-side cookie is written for keeping track of the session. Once validation is done, the frame in the browser is redirected to another XML document specific to that user (username.xml). That document is created the first time the user logs in and is afterwards updated with details of the previous orders, giving quick and direct links to those products.

5.4 The XML Database

Le Gastronome sells several 100 products, but the data that this represents fall neatly within the hierarchy of categories and subcategories mentioned previously. As the range grows, it can even be broken down further (e.g. white wines could be categorised further into sweet and dry whites). No XML files on the site are bigger than 15Kb for efficient download. The file at the top of the hierarchy (MENU.XML Fig. 1.0) contains a list of the different product categories available on the site, with reference to the actual XML files where category information is stored. A Document Type Definition (DTD)
exists for the product files, this to ensure that the files are valid, i.e. they conform to the required structure for the system before being put 'live' on the website. This is used by the maintenance utility, as current web browsers do not validate XML documents.

5.5 Presentation Layer

This layer is accomplished of course with XSL style sheets, with the addition of some scripting (JavaScript) for navigation and validation, and some Cascading Style Sheets (CSS). XSLT is used for styling the structure and contents of the pages (i.e. what goes where); CSS is used to apply a consistent format for fonts, table cell headers and colours, etc... this separation of functions for the two style sheet types simplifies maintenance, and used this way XSLT and CSS complement each other brilliantly.

There are only three XSL sheets for the whole site (Figure 1). The first one renders the categories XML file in a top frame, formatted as a menu bar and also a drop down box from which users can use to navigate to the various product categories. A design goal is that no product listing should be further than three clicks of the mouse. The second style sheet is applied to each and every product XML file, since they share the same structure. Any functionality required (e.g. displaying a photograph, sorting, order of columns, etc...) can be added to the XSL document and becomes available to all the product pages; this greatly simplifies the maintenance and update of the presentation logic. The third sheet is used to display the user specific home page upon login. The combined style sheets weigh 8Kb in total; the total documents for the site loaded at any one time is no more than 25Kb, on top of which some product photos will be loaded as per design. This makes the whole site remarkably fast loading, efficient, and enjoyable to browse even over a slow modem line.

5.6 Secure Contents

Some of the site contents depend on the channel status of the logged user. The logic in the products style sheet will read the user status from the cookie set at login and render the product information accordingly. Pricing information is available via a CGI script which responds to channel and product id parameters. This was done so that pricing information will never be downloaded as a file to the client browser (a requirement set by the company manager).

5.7 Maintenance

The advantage to a company of having a data driven web-site would be naught if there was no capability of updating the data in the XML database. Although one can open any XML document in Notepad, there is a need for more complete tools to assist in that task (Sharpe, 1999). Currently, the data is created and updated in MS Excel, then saved as comma separated files and converted to XML using an ad-hoc C++ conversion program. The site is then updated via FTP. The future plan is to create an online administration utility, possibly using the same PERL technology as the secure components of the site.

This website as described here has been built as a prototype. The challenge of course is to see whether the idea can work under the harsh commercial reality. A more sophisticated version of this site, still using the same technology is now being implemented. You can visit it at: http://www.legastronome.co.nz. Although some changes are being made to the design expressed here, they mostly relate to changing commercial requirements, as can be expected in a prototyping development process. The cross-browser compatibility problems have also been resolved. What has struck the author, a developer experienced in expensive database and web technologies (Oracle), is the ease and speed by which layout and even process changes can be achieved, once familiarity with the technology is attained. But this is, as they say, another story for another day...
6. Conclusion

It is possible to create a fully functional e-Commerce Web site using nothing else than XML and related technologies. The open-standard nature of XML and XSL adds the advantage of nil licensing cost for such a site and offers easy and total portability of the web site framework and contents. This frees the e-Commerce entrepreneur from being dictated by the hosting service provider (ISP) what technology (e.g. database) is made available and at what cost. It means a much greater choice of potential hosting services, and no 'technological' lock-in with a non-performing service provider. Additionally, it only requires a two-tier architecture, and gains another great boost to performance by effectively distributing the processing requirements of the presentation layer.

This type of architecture is certainly not suitable for each and every e-Commerce venture; in some cases, security and data structure considerations may make it impractical (e.g. it definitely would not be suitable for an online banking site!). It does make a viable alternative to proprietary technologies like PHP, ASP, and other N-tier architectures commonly in use today. It is especially suited for the budding e-Commerce company who wants to retain a high level of control and portability of their web contents. With the advantages to business described here -especially, but not only, to small and medium businesses - the author believes that the change is inevitable and that more and more web sites in the near future will be created using XML and related technologies. This will require a pool of people skilled in these technologies to create and maintain such systems. Our challenge and also our great opportunity remains to train them.

References

Harold, R. E., 2004, Effective XML. Boston: Addison-Wesley
Kennedy, D. 2003, Relaxing with XML data structures. In "Proceedings of the 16th

Copyright © 2005 Jouvernaux, M.

The author(s) assign to NACCQ and educational non-profit institutions a non-exclusive licence to use this document for personal use and in courses of instruction provided that the article is used in full and this copyright statement is reproduced. The author(s) also grant a non-exclusive licence to NACCQ to publish this document in full on the World Wide Web (prime sites and mirrors) and in printed form within the Bulletin of Applied Computing and Information Technology. Authors retain their individual intellectual property rights.

Copyright ©2005 NACCQ.

Krassie Petrova, Michael Verhaart & Christo Potgieter (Eds.)
An Open Access Journal, DOAJ # 11764120 , (☑zotero)