Structure in documents: an introduction

Being an introduction to the use of databases and markup languages to help the designer make electronic and paper documents work harder and be more useful, concentrating on the markup language called XML.

Visible and invisible structure

Visible structure: regular, systematic use of type, space and colour to expose a document's meaning to a human user. Codified in

- grids
- style manuals

Invisible structure: regular, systematic use of electronic markers to expose a document's meaning to a machine. Codified in

- database tables and schemas
- SGML/XML Document Type Definitions and schemas

Why is structure important?

A structured document allows us to:

- design sites rather than pages
- publish different ‘slices’ of data for different audiences
- auto-publish new data
- navigate documents according to their meaning
- exchange information between systems

Two overlapping structural systems

Markup languages such as XML and SGML add labels about structure within documents . The labels are often directly human-readable.

Databases separate different types of information into different fields within an overall structure. The labels exist within the database software rather than the document. In fact ‘document’ is an alien concept to most databases.

Why XML matters

XML is the replacement for HTML – it is the new language of the Web

XML is a way of exchanging information between many different computer systems

XML is the mandatory standard for the exchange of information between government information systems

What XML is

A standard way of marking-up information

For publishers: a way of separating content from appearance

For IS professionals: a way of exchanging data between different computer systems.

How it works

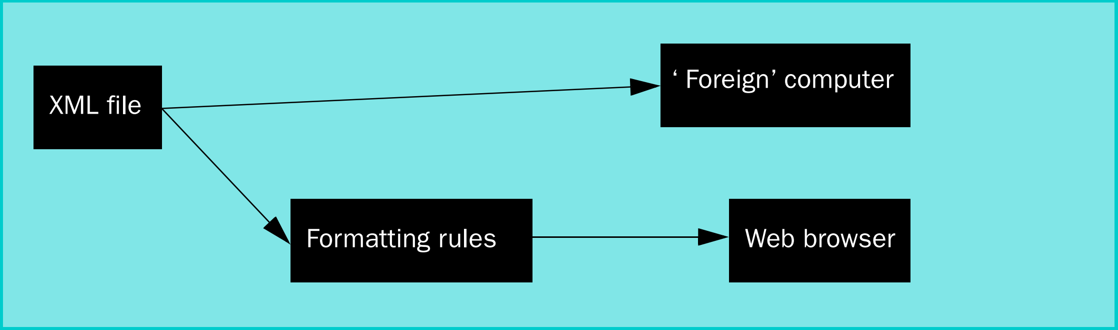
XML and HTML

XML is a *markup language* like HTML.

Like HTML, XML can be used (via separate formatting rules) to control the typography of a web page – but with greater control and flexibility.

Like DHTML, XML can be used (via separate formatting rules) to control the layout and behaviour of a web page – but much more powerfully.

Unlike HTML, XML is ideally suited to move information directly between computer systems across the web.



HTML describes The *appearance* of information – text and pictures

XML describes The *structure and relationships* of information – text and pictures and data

What XML looks like

XML – like HTML – works by surrounding text, pictures and other information with ‘tags’:

<tag>This is some text</tag>

Tags in HTML

In HTML there is one set of tags – a single *schema* – to cover everything on every web page throughout the world:

<H1>Heading</H1>
<P>And a paragraph</P>

Tags in XML

In XML there is a set of tags for a particular class of documents: there are many *schemas*, each specific to a particular class of document:

A schema for invoices might have tags called <item>, <price>, <quantity>, etc

A schema for books might have tags called <prelims>, <body>, <chapter>, etc

Schemas in XML

A schema describes a class of document or a class of information.

A schema says what will be in all documents (*instances*) in the class.

Schemas allow computers to talk meaningfully to each other

Almost all XML documents conform to one (or more) schema

Some schemas are public and world-wide (see http://www.schema.net/), some are private, local, or confined to a particular industry or interest group or project.

Making or using the right schema is critical to the success of any project.

Putting XML on the web

How XML reaches the web

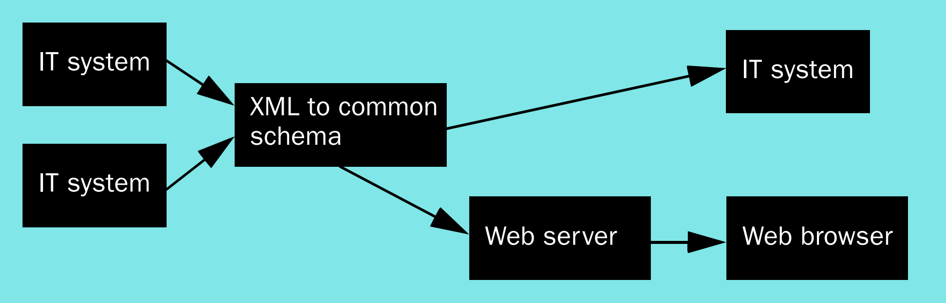
Today, very few web sites are ‘published’ direct in XML.

Instead, if XML is used, it works ‘behind the scenes’ – the web server reads the XML, transforms it to HTML, and sends the HTML to the user.

XML is mostly used today as

- an intermediate format joining IT systems to the web
- a way of delivering ‘parallel publishing’ projects

Joining IT systems to the web



Websites which knit ‘live’ IT systems to the public or other businesses use XML as a ‘system-neutral’ way of sending and receiving information.

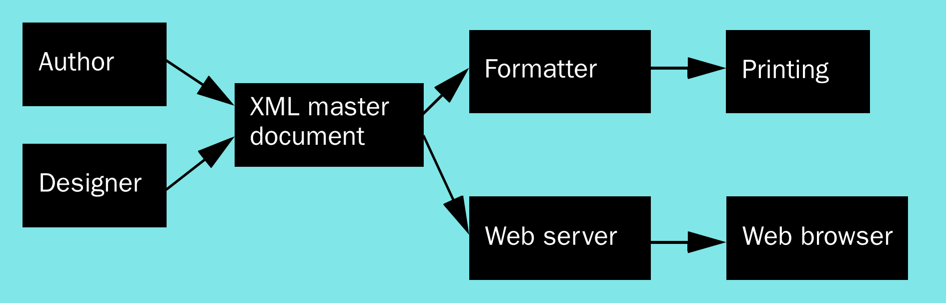
For IS professionals

- ‘Middleware’ to create XML from traditional IT ‘back-end’ systems is easy to write
- XML can stay the same when ‘back-end’ systems change
- the web can be used to send XML

For web managers

- Designing for XML is simpler than using ASP/JSP etc
- Web servers increasingly understand XML
- Servers can transform XML to different HTML flavours on the fly

Delivering 'parallel publishing' projects



Projects where the same information must be available in print, web, and possibly other formats at the same time.

The same XML 'master' document can be printed (via a PostScript formatter and conventional print) and published via a web server or content management system.

The ‘look and feel’ of the documents can be very different – design and interaction are built-in to the PostScript formatter and web server without changing the XML master document.

Direct to web

Most users have XML-aware browsers but they are not yet common enough to allow ‘direct’ XML publishing, except for specialist/controlled audiences.

XML-compatible browsers include:

- Internet Explorer 5.0 or higher
- Netscape Navigator soon (Version 6 Preview 1 or higher)
- Opera 4.0 or higher

XML and design

XML documents generally contain no information about design – just information about the kind of content.

Design is ‘applied’ by a formatting system (for example, a web server or sometimes a web browser) which uses formatting rules, just as HTML uses Cascading Style Sheets (CSS).

The formatting system says ‘make this kind of information look like this on the web.’

Standards for formatting rules

- CSS – Cascading Style Sheets
- XSL – eXtensible Style Language
- XSLT – eXtensible Style Language and Transformation

Each is more powerful than the last.

CSS

Cascading Style Sheets are familiar to web designers now, and work well with XML as well as HTML files. The standard:

- is stable
- is firmly focused on the web
- allows quite good control of layout and typography
- works with HTML in the browser
- is a ‘transition’ standard towards XSL/XSLT

XSL & XSLT

eXtensible Style Language and eXtensible Style Language – Transformation are new standards which work only with XML (and XHTML) files. They:

- are not quite stable standards
- can be used to control design on the web and also (soon?) for print.
- allow excellent control of layout and typography
- allow selection and ordering as well as design
- work with XML on the server and send transformed HTML or XML to the user.

Structure in databases

The traditional database consists of tables of information in which rows are called records and columns are called fields.

	name	**size**	**colour**
record1	Fred	large	green
record2	Natalie	small	orange
record3	Fiona	large	blue

Most databases these days are relational: one database uses many tables with relationships. There are other types.

Talking to databases

There are no standards for the internal working of databases, but there is a widely-supported language – SQL (Structured Query Language) – for putting information in and getting it out of the database.

You can use SQL (or application-specific tools) to add markup to database output.

You can use ASP or JSP or PHP to call information from a database with suitable drivers.

Markup and databases

Most databases don’t understand markup – either procedural markup such as HTML or PostScript, or structural markup such as XML or SGML. It’s

- (quite) easy to get databases to output marked-up documents.
- herder to get them to read marked-up documents, though this is changing with the widespread adoption of XML throughout information systems.

Database-centred publishing systems

Two main technologies used in database-based publishing systems:

- Relational or flat databases
- Object repositories

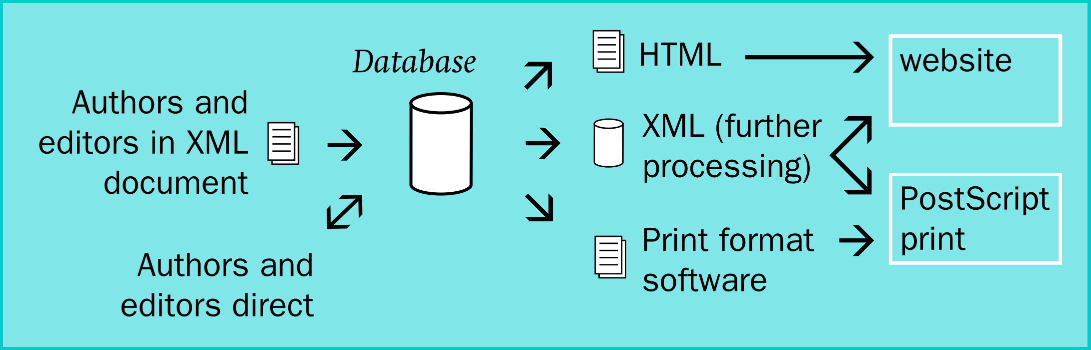
Conventional database publishing

Database used as the master format, surrounded by input (editing) and export (publishing) filters, or exporting to suitable publishing software

Oracle, Informix, IBM DB2 and others are relational database management systems (RDBMS) used in many web-publishing systems such as MediaSurface and StoryServer.

Lotus Notes/Domino is a ‘flat’ database and web publishing system.

Database-centred publishing systems *Conventional database publishing*



Database

Authors and editors in XML document

Authors and editors direct

HTML

website

XML (further processing)

Print format software

PostScript print

text matters

Plus points

- ideal for ‘table-based’ information such as catalogues, price lists and statistics
- straightforward to build web-forms for authors/editors
- straightforward to export HTML and/or XML
- easy to make ‘live transformations’ on data reflecting changes instantly
- well-understood by IS professionals

Minus points

- Not suited to ‘document-like’ documents (!)
- Output to print often difficult: bespoke formatting software or separate print-publishing process may be required
- Remote authors need live web connection
- For most purposes, web-form authoring is unpleasant

Object repositories

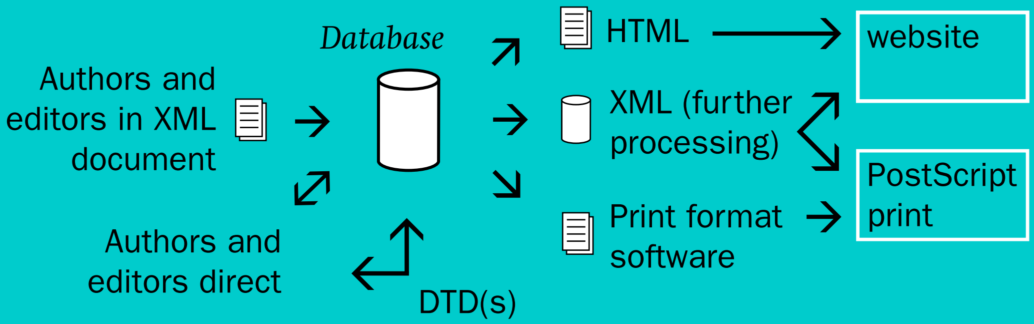
Object database used as the master format, surrounded by input (editing) and export (publishing) filters, or exporting to suitable publishing software.

Object databases deal with ‘tree’ data structures as well as ‘table’ data structures. This is useful in dealing with XML.

Normally contain ‘workflow’ tools for revision control

Oracle IFS, Zope, Frontier, POET: x-hive

Database-centred publishing systems *Object repositories*



Database

Authors and editors in XML document

Authors and editors direct

DTD(s)

HTML

website

XML (further processing)

PostScript print

Print format software

Plus points

- Object databases are cool
- System design is simple – in theory
- Usually suited for workflow management
- Work well with ‘document-oriented’ XML

Minus points

- Not many products to choose from
- Can be expensive
- Tools are scarce or young, therefore
- Implementing projects can be risky/expensive

Who does what

Web publishing already involves designers and IS professionals. They will have to work increasingly closely because:

- XSL and XSLT are too complex for most designers
- user interaction and branding are too complex for most IS professionals
- e-everything is getting more important: getting everything right is more and more important (and harder and harder)