

Single Sourcing at the STC 50th Annual Conference

Bonnie Yelverton

As I wrote in my first article on the STC 50th Annual Conference last month, I spent most of my time at presentations concerned with SINGLE-SOURCING . In this article I review the five sessions I attended on this topic. Most of the materials for the conference have been uploaded to the conference website at: www.stc.org/50thConf/sesMaterials.asp. *(Right click on the links and save them to your computer. I have not been able to open them on-line.)*

Copyright by Bonnie Yelverton - Upland, California, June 2003

About Single Sourcing

SINGLE SOURCING means outputting several different documents from a single source file. The resulting documents can be of different file types, such as a printed manual, pdf, an HTML Help file, a web page or even a WAP file. Or the documents can be various versions in the same format aimed at different user groups.

Single Sourcing to Various Formats

There are many different TECHNICAL SOLUTIONS to Single Sourcing.

- Saving a document as pdf is one of the earliest methods, providing an easy way to display documents on the Internet.
- Help files created in Robohelp can be saved in a number of formats, including as Word Documents.
- WebWorks enables you to output a FrameMaker or Word document as an HTML Help file.
- You can save a WORD or PowerPoint file as HTML.

With these methods, you output exactly the same content to different file formats for different uses. If you want manuals for different purposes, you still have to write separate manuals, which often means that you have to disturb the SUBJECT MATTER EXPERT several times for each version. If the manual is HTML, the user can search for specific topics, but there is only the one treatment of a topic available in any one manual.

XML and Single Sourcing

Single sourcing is one of the major reasons tech writers get interested in XML. XML provides much more flexibility to single-sourcing. With XML you structure a document, enabling you to output different sections to different documents. For example, you can identify sections for beginners, intermediates or advanced users, for different security levels, or for different tasks. The writer prepares the same topic in different ways for different users and file formats at the same time. Only one writer needs to talk to a SUBJECT MATTER EXPERT about a given topic once, rather than different people asking him the same questions for different documents. This is a great argument to use with management to get them to go over to XML-based documentation, as it saves a lot of time and resources.

The structure is specified with a DOCUMENT TYPE DEFINITION ([DTD](#)) or XML-SCHEMA, which is established prior to writing the document. There are a number of standard DTD's and Schemas available that are generic enough to be used for many different applications. Or you can create (or have someone create) a special DTD for your purposes. A DTD defines the ELEMENTS and the hierarchical, nested element structure that can be used within a document type, as well as the ATTRIBUTES that can be used to further specify and Element.

The completed XML document is outputted to the various formats with XSL -STYLESHEETS, which are also XML documents. These can be coupled with .css stylesheet formatting for webpages and other formats.

The tech writer's favorite authoring application, FRAMEMAKER has long had a version for producing SGML, the forerunner of XML. The new Version 7 enables you to write so-called STRUCTURED DOCUMENTS quite simply. FrameMaker uses [templates](#), which include ELEMENT DEFINITION

DOCUMENTS , combining a DTD with styling rules, and other properties. These are usually set up by an application developer. Adobe is pushing Structured FrameMaker as an easy way to revise documents, since everything that belongs together is a single ELEMENT. That way it's easy to move things around, or have different people working on the same document and adding things together.

There are lots of other "light" AUTHORING TOOLS like XMetal, Authentic, which comes with my favorite XML development tool XMLSPy, and Veredus from Rascal software, which I heard about at the Conference. These tools make it easy for writers who don't really want to know much about coding XML to do it anyway within a WYSIWYG environment that can even be set up with STYLESHEETS so you can see what it's going to look like. This is creating a whole new job market for those of us who are interested in the "under the hood" stuff, to create [DTDs](#) and XSLTs .

Levels of Structuring

This table shows the levels of structuring which are used in different types of Desktop Publishing applications. It is taken from the page "Analyzing Document Structure".

Table contents
Headings
Type
Example
Characteristics
Contents of Rows
Unstructured documents
QuarkXPress document
Look and feel. <i>"It's all presentation. No points for technical merit."</i>
Semi-structured documents
FrameMaker or Word with consistentlytagged styles.
Look and feel. <i>"Use tags to organize the look and feel. Presentation counts, but tagging is important."</i>
Structured documents
FM Structured Document, XML
Determine hierarchical tagging structure based on content. Formatting based automatically on structure.

A Single Sourcing Case Study

TIMO HAPPONEN and VESA PURHO from the Finnish Telecommunications giant, NOKIA , told about their efforts to convert [Nokia](#)'s enormous documentation to a SINGLE-SOURCING system. There are handouts from both of them on the STC site.

Nokia Networks Documentation with Re-use Dimension

Nokia is not just innovative cellphones, it is also a global mobile network provider with hundreds of products. The customers and users for this documentation are:

- mobile networks operators
- service providers
- technical experts

who need documentation of from 100-30,000 pages (requiring several cartons of binders to transport.), which are now published mainly as html, pdf or even paper. They started to get away from paper documentation after finding the binders in a corner still in the shipping cartons. Nokia is transitioning from SGML to XML, and uses mostly Adept and Epic as authoring tools.

Nokia was motivated by several dimensions to use SINGLE SOURCING:

Information re-use dimensions

- Being able to use multiple publishing media
- Reusing material when multiple products based on the same platform have large amounts of common software.
- Combining system level and product level documentation.
- Providing documentation for multiple audiences.

Business drivers for re-use

Table contents

Headings

Improvement

Area

Contents of Rows

Reduction of COSTS

CONTENT creation, maintenance, management, publishing

Shortened cycles

Content creation, maintenance

Increased responsiveness and flexibility

Changing product and customer needs

Increased consistency and accuracy

Quality, usability, look&feel

Information Model

Nokia bases its INFORMATION MODEL for single sourcing on the overall TELECOM OPERATIONS MAP (TOM), which was developed by the TeleManagement Forum. The TOM defines the core

business processes for a telecom operator. Nokia has further refined this as the Nokia Operations Model (NOM), which consists of managing:

- Business
- Customers
- Services
- Network

Each of these areas requires documentation, which is further defined in the Information Model. The Information Model is necessary to plan the documentation structure, but Nokia has learned a number of lessons along the way in this connection:

- The model must be based on real user information, but it is not the whole truth about the users.
- All relevant product areas and variants must be included to get wide commitment to the process - but in a widely diverse product environment, the solution must be a compromise - and an on-going process.
- Be as practical as you can.
- Involve customers where possible

One thing they found was that there is a great need for STANDARDIZATION of the information structure in the telecom industry, because of the horizontal nature of networking. Therefore, Nokia has invited interested companies to participate in this process.

Single-sourcing components

Nokia is implementing single sourcing by using modul documentation, as described in the following table..

Table contents
Headings
Abbreviation
Type
Description
Contents of Rows
MC
Media Configuration
A list of DUs and MUs belonging to a certain information product.
MD
Media Distribution
A frozen media configuration, used for publishing purposes.
DU
Description Unit
Collects a set of MUs into a paper document, with cover page, disclaimer, TOC, etc.
MU
Manageable Unit
Independent piece of information that is listed in a TOC. Can include SUs.
SU
Storage Unit
Reusable piece of information (graphic, text, etc.) that only has meaning as embedded in an MU.

Be sure to download the handouts for this presentation from the STC website, as there are some excellent diagrams to further clarify the system they developed, as well as more discussion of the lessons learned so far in their implementation of single sourcing.

Analyzing Document Structure and Creating Structure Definitions in FrameMaker 7

SARAH O'KEEFE, author of *FrameMaker 7: The Complete Reference* and other books on technical writing, described how to create FRAMEMAKER ELEMENT DEFINITION DOCUMENTS , the FrameMaker [DTD](#) application. See her white paper at www.scriptorium.com/structure.pdf.

Document Design Types

Sarah divides documents into three types based on their level of structuring.

Table contents
Headings
Type
Example
Characteristics
Contents of Rows
Unstructured documents
QuarkXPress document
Look and feel. <i>"It's all presentation. No points for technical merit."</i>
Semi-structured documents
FrameMaker or Word with consistentlytagged styles.
Look and feel. <i>"Use tags to organize the look and feel. Presentation counts, but tagging is important."</i>
Structured documents
FM Structured Document, XML
Determine hierarchical tagging structure based on content. Formatting based automatically on structure.

If you are accustomed to working stringently with paragraph styles in FrameMaker or Word, moving on to XML is conceptually not too difficult. With structured documents, nothing is "Normal" anymore. You can use any tags you want in your document structure. Everything in the text is tagged hierarchically, so that it can be demonstrated in a tree structure. The tagged sections are known as ELEMENTS . Elements can be further specified with ATTRIBUTE s, which are called METADATA in FrameMaker.

Designing structure templates

When designing TEMPLATE s, it is very important to *"Plan from the top down, but implement from the bottom up."* Analyze your material and divide into into a reusable hierarchical structure, where CHILD ren elements are nested within their PARENT s as in the following structure for an article or book:

- Chapter
- Section
- Paragraph
- Emphasis

A chapter has sections, which contain paragraphs, which may contain words that are marked to be emphasized. Note that both a chapter and a section could have a title CHILD, and that a chapter could have several sections, and a section several paragraphs (or even sections). Everything nests within

each other. You can use the same Title and Section definitions, even if they are used in different levels of the HIERARCHY. You might want to diagram the structure first. Think of all the different instances of your document type to make it as generic as possible.

The tricky part of a structure template is deciding which METADATA (ATTRIBUTES) to use to further define the elements. In some cases you might want to use sub-elements, in others, attributes. For example, a document type may have a number of sections that are always the same, for example, *Concept, Procedure, Example, Summary*. You could make a very specific DTD that has separate elements for each of these types. Or you could have a structure where the Section elements have an attribute called "content", which can be specified to be one of the four types. Each application is different. It is up to you to decide what level of structure you want. *"Add metadata when communicating additional information (often formatting) not required by the structure, or to add another dimension to the content."*

Building EDDs

When you have thought out your structure you can build the EDD. This is done in three steps:

- Build and test the structure
- Add the attribute definitions.
- Add formatting information

The last item requires a decision. If you include formatting information in the EDD, you limit its usefulness. If you only expect to use the document in a single format type, such as printed, or HTML, then it is a possibility. But to keep your options as open and flexible as possible, Sarah O'Keefe recommends that you use separate formatting templates for the different applications. This provides much easier maintenance of the formatting and separates structure architecture from formatting, which may also be done by different designers. It is rather the same distinction with using inline formatting in HTML or separate .css stylesheets for a whole site.

Sarah O'Keefe concluded her session with a question: *"What are the three most important considerations in building structured templates?"* She answered herself:

- Planning
- Planning
- Planning

FrameMaker-XML Round Trip

MARY ANN HOWELL told about her project to learn how to use XML and FRAMEMAKER. She spent almost a year of underemployment working on learning structured documents and creating EDDs. She worked out a method to “*chunk*” an XML document into separate sections that can be stored in a database. The completed XML document is created with an XSL stylesheet with includes, which pull different chunks together from the database. Download her materials from the STC site as well as from her website: www.hikaripub.com, where you can also download a little CHUNKR program to chunk an XML document up into bite-sized pieces for storing in a database.

Adobe's Examples and Tutorials for XML and EDDs

Adobe provides you with a number of examples of structured documents that you can work through on your own.

- The XML Cookbook - at Program Files / FrameMaker 7.0 / XMLCookbook
- EDD in the Cookbook - at FrameMaker 7.0 / XMLCookbook / Completed
- FMSGML EDD - at FrameMaker 7.0 / Samples / FMSGML
- How-to for structured documents - at FrameMaker 7.0 / OnlineManuals

Mary Ann Howell's guide to creating a chunked document

The following is her outline for turning an unstructured document into a structured FrameMaker document. For the details, please consult her handouts. She recommends keeping a notebook on hand to record how you solve problems as you go along.

- Make a structured FM document
- Translate the structured FM doc to an XML doc
- Make XMLchunks and link them to the structured FM document as XML insets.
- Edit the XML chunks

XML Under the Hood - prototyping web interfaces and simple apps without DTDs

DOROTHY J. HOSKINS showed how to use XSLT transformations with a simple XML document to prototype a web site. Her handout and other materials are at: www.resourx.com.

Light Weight XML Apps

Dorothy Hoskins explained how to create LIGHT-WEIGHT XML APPS with DTDs for quick prototyping in a short development timeframe. You can create different modules (for example in different languages) and try them out in a simple user interface, to determine the final document structure. This is particularly good for multiple USER-SELECTABLE VIEWS, topic maps or similar tools.

The web as an interactive interface

“Instead of page-for-page redition, you can create numerous presentation views” for different user groups and needs. Users can create ther own view by selecting details in drop down boxes and the like. It is even possible to track USER INTERACTION, which can be used further in design considerations.

Creating Graphics for Both Web Pages and PDA Displays

SARA KUBIK introduced us to the world of SCALABLE VECTOR GRAPHICS (SVG), which are actually XML documents that can be accessed both by XML and XSL to make dynamic changes, such as call-outs for diagrams. She has several handouts on the STC site.

Raster, Vector and SVG Graphics

Currently most graphics on the web are RASTER graphics, such as .gif, .jpg and .png files, which consist of “*a RECTANGULAR grid in which different pixels illuminate specific colors.*” If you enlarge a raster image, you will clearly see the separate pixels. Nevertheless raster graphics are excellent for displaying the texturesm feathering, and blurring of photo-realistic images.

VECTOR graphics are “*smart*” , in that they are “*created through mathematical commands that define their size, position and geometry,*” in other words, shapes, lines and curves. They are great for images that don't have to be realistic, but that need to be scalable without losing detail. Typical uses are for maps, Computer Aided Design, such as Visio diagrams, Flash animations and Illustrator graphics. Unfortunately, there are many different file types, which must be either converted to raster graphics or used with a VIEWER to display outside the original program.

SVG graphics are similar to Vector graphics, but they are defined as an XML document, which means that they can be manipulated just as any ther XML document with XSLT and other XML technologies. You see SVG graphics at work in, for example, web weather sites, which show varying icons with sun, clouds and rain, depending on the weather data they are based on, or car sites, where you can "design" your own car. Since they are SCALABLE, the same diagram can be used for many different screen types, including those used to illustrate this section. The biggest drawback right now is that they are not currently supported by the latest versions of INTERNET EXPLORER and NETSCAPE. You still need a VIEWER to display them. If you'd like to be able to view SVG images, you can download the SVG Viewer from Adobe:

<http://www.adobe.com/svg/viewer/install/main.html> See also the demos on the SVG home page, while you're there. However, some of the newest cellphones made by Nokia, Eriksson, Texas Instruments and some Japanses manufacturers, have it installed free, and Pocket PCs (but not Palms) are also supported.

SVG for the Programmer

Since SVG is XML, PROGRAMMERS can integrate SVG diagrams with the other the other XML lanuges, stored in databases and used in networks. They are open-source graphics that don't need expensive software to create them (although some of the expensive software, like Illustrator, can be used to create them.) If you right click on a SVG graphic, you can view its source code - and borrow it - just like an HTML or XML file.

SVG for the Designer

The DESIGN community isn't exactly eager to get into SVG graphics because of their OPEN-SOURCE character. It is rather difficult to protect your work. Nevertheless, the Vector based graphics programs are beginning to add SVG facilities. SVG have so many advantages, it is hard to ignore them for long. Among the advantages:

- Potentially faster downloading than raster and vector files, since they are just text. They

- don't even have to be reloaded when you zoom in on them.
- Search engines and software like Jaws can read and rank them.
- Greater number - potentially 16.7 million - colors available, compared to the 256 websafe colors available for raster images colors.
- Not really a competitor to Flash, which is best for high-end stand-alone animation. You need both tools to be effective.
- Used for simple, non-photorealistic designs that need to be resized or refocussed without reloading, such as floor plans, schematics, exploded views.
- Use for simple, possibly interactive animation, sch as charts based on information from a database, but use Flash for complex applications.
- Use Illustrator, Corel Draw, SVGMaker, CAD2SVG or even Notepad to create them. FrameMaker can import SVG to structured documents.

SVG for Business

Since SVG files are vectors, you don't have to pay designers for various versions of the same graphic.

- To change call-outs, you just change the file (or hook it to a database with, for example, different languages).
- SVGs maintain their clarity no matter what size they are shown (or even if they are skewed), so one size fits all.

Glossary

ATTRIBUTE : Special nodes that are added to elements to further specify the element. For example, you can specify user or security level with attributes.

Chunk: A section of XML, which can be stored in a database and then included into a larger XML document. Used to make flexible documents for various users and needs.

DOCUMENT TYPE DEFINITION : The hierarchical list of XML elements and attributes use to structure an XML document.

ELEMENT DEFINITION DOCUMENT : A FrameMaker file containing the rules for a structured FM document. Defines not only which elements are legal, in which order, as with a [DTD](#), but also defines the formatting of the text, in the same way as [XSL](#).

ELEMENT : The basic building block of XML, which are defined in a hierarchical structure. All sections of the document are enclosed in nested element tags.

Raster Graphics: Type of graphic based on a RECTANGULAR grid in which different pixels illuminate specific colors. Typical formats: .gif, .jpg, png, bitmap.

READ/WRITE RULES : Translation between the FrameMaker [EDD](#) and the XML [DTD](#).

SINGLE SOURCING : Using a single source file to output different documents, in different file types and/or content.

STRUCTURED APPLICATION DOCUMENT : A FrameMaker file which serves as a pointer to the support files (such as the [EDD](#) and [Read/Write Rules](#)) used during the conversion to XML.

SVG : Scalable Vector Graphics

TEMPLATE : A blank FrameMaker document with formatting and element definitions ([EDD](#)) attached.

Vector Graphics: Type of graphic created through mathematical commands that define their size, position and geometry, in other words, shapes, lines and curves. The many different file stypes, such as CAD and Visio diagrams or Flash animations, require a viewer outside the software that creates them.

XML : Extensible Mark-up Language. XML marks up text according to meaning, not output format. Thus sections are marked according to their content, and can be further defined for different purposes. One xml file can include sections about the same issue intended for different output methods. When XML files are coupled with [XSL](#)-Stylesheets, they can be transformed into various versions. Please see my earlier article about XML in the [September issue](#) of *Ruff Draft*.

XSL : Extensible Stylesheet Language is an XML document which is used to transform XML to another form, either another XML document, HTML, WAP, pdf, text or even [SVG](#). It has several forms: XSLT for transformations to other file types, and XSL-FO, for formatting print text.

Index