

# XML and Content Management for Instructional Design

Henry Meyerding

July 2003

## **Abstract**

XML as a content reuse methodology is a very complex and technical topic. Instructional designers who seek to discover the impacts of XML to instructional design very often are frustrated by the technical white papers available on this subject. Those white papers usually fail to speak to the practical needs of instructional design. This article is written for experienced instructional designers who wish to understand content reuse in an XML environment. It explains the basic underlying concepts at a non-technical level and relates those concepts to instructional design processes. The effects of content reuse are described for both print and online curricula. Return on investment and value to the organization are examined.

# 1 Introduction



*Write it once ... use it many times.*

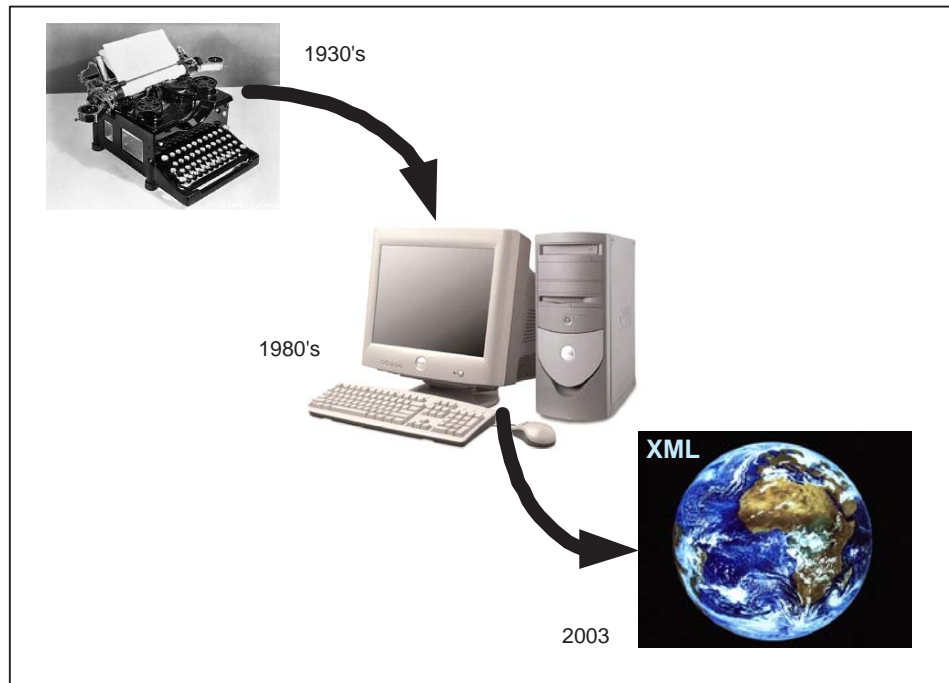
This simple statement describes the function and attraction of XML as an authoring methodology. This paper discusses XML and how XML can be used in an instructional design setting to manage and facilitate the definition and use of learning objects.

In its simplest form, a learning object is some discrete particle of knowledge that addresses a specific learning objective. That being said, learning objects are not simple things, but complex constructs of information, presentation and interaction. Designers traditionally have seen themselves as artisans who created unique learning tools for each new learning situation. They have been very slow to transition their thinking to a systematized approach to the development and delivery of learning.

At the same time, the enterprises for which most instructional designers work have been under increasing pressure to provide training more efficiently. Training departments, unwilling or unable to deliver substantial increases in efficiency, risk being replaced by outside contracting firms that promise that they can deliver these efficiencies.

XML has become a standard means of information interchange within the computer industry. Using XML to create and manage learning objects is a proven technology that offers greater efficiency without reducing the quality of the training deliverables.

Working within an XML environment requires a change in perspective. Instead of approaching each task as the resolution to a specific obstacle to learning for a specific audience, the designer must analyze the task in a wider context. With whom does this audience share this learning requirement? How can this learning obstacle be resolved for all students? If this objective does not apply to other groups, are there components within it that do apply to a wider audience? What existing training can be pulled into this task and modified to work, without affecting the quality of the learning? What other training is relevant to the content currently under development? How can these new content objects be fitted to other uses?



Instead of being a creative artist fashioning unique responses to specific learning requirements, the designer becomes a production professional who analyzes the learning needs of a specific group as those needs relate to the generalized requirements of the entire learning community. The importance of designers understanding the capabilities and rationale behind the content reuse system cannot be stated strongly enough. There never has been a system that so good that it could not be rendered totally ineffective by resistant participants. Managers must be evangelists of XML technology's liberating capabilities. They also must be zealous in training their staffs to understand these capabilities.

One of the incidental benefits of operating in a learning object environment is that designers are exposed to content created by other designers much more than in traditional project environments. Properly managed, the specialized understanding of different teams is more effectively shared and the quality of the output is increased[9].

## 1.1 Terminology

Before launching into a discussion of XML[1], it is important to understand some of the terminology that will be used throughout this paper.

**Attribute** - The characteristic of an XML element that defines the content. Example: If the elements are class, type, and color; corresponding attributes might be toy, rubber ball, and red.

**Chunking** - The process by which legacy content is tagged for inclusion in the content database.

**Content** - Content is information. It may take the form of text, graphics, audio, or video.

**Database** - A hierarchical distribution of data arranged in relationships that provides quick access to information of interest.

**Document** - Strictly speaking, when working in XML there is only one super document that contains all the content. This content fits into a common structure. We extract pieces of this super document and publish it as document instances, which may be either static or dynamic.

**Dynamic Instance** - When publishing a document instance, it may contain information that changes continuously. In defining the document publication instance, it may be desirable that every time a user opens the instance they see the most recently updated information. Contrast with static instance.

**Element** - An XML element is a definition for content. Any piece of content may be defined by one or several elements. Example: class, type, and color.

**Learning Object** - The base component of training curriculum. Each learning object addresses a specific learning goal. Just how specific a goal is defined varies from system to system.

**Metalanguage** - The language that is used to talk about (expressions of) another language, the object language. XML contains and identifies content, but the XML is not the content.

**SCORM** - Sharable Courseware Object Reference Model - A set of specifications for developing, packaging and delivering high-quality education and training materials whenever and wherever they are needed.

**Static Instance** - When publishing a document instance, it may serve as a standard or reference. In defining the document publication instance, it may be desirable that users see a single, unchanging document, until any changes have been approved by a ruling/governing body. Contrast with dynamic instance.

The next section introduces XML and discusses some of the features of XML that make it particularly appropriate for learning object development and learning content reuse.

## 2 XML Basics

What is XML? The typical definition of XML:

XML is a World Wide Web Consortium (W3C<sup>1</sup>) standard for Internet markup languages. XML is called "extensible" because it is not a fixed format like Hypertext Markup Language (HTML). It is designed to enable the use of Standard Generalized Markup Language (SGML) on the World Wide Web. It is an abbreviated version of SGML, making it easier to define document types, and to make it easier for programmers to write programs to handle them. It omits the more complex and less-used parts of SGML in return for the benefits of being easier to write applications for, easier to understand, and more suited to delivery and interoperability over the Web.

This definition provides a lot of information about XML, but it was obviously written by someone who lacks an understanding of what XML is and what it is for. XML was designed as a database metalanguage[5]. It was designed as a means to structure content so it could be put into and be retrieved from a database in a form that was useful for content reuse. Information content can be text, graphics, audio, video, or complex constructs of all these learning components.

The principal difference between XML and HTML is that the former uses "smart tags." Smart tags convey information about the content they contain. Because of this, you can use the structure you create to put your content into an easily retrievable form. One exciting aspect of XML is the ability to define your content your way, creating custom tags for different kinds of instructional objects such as objectives, test questions, feedback and all the other common components of training content[3].

Your content management system reads the smart tags and parses your content into useful chunks that you can assemble into subsequent documents. When you need the same, or similar content, you construct a query of your content database. You then review the resulting content and if it matches your current use, you use it. If you find nothing useful, you add new content for your current document – and future use.

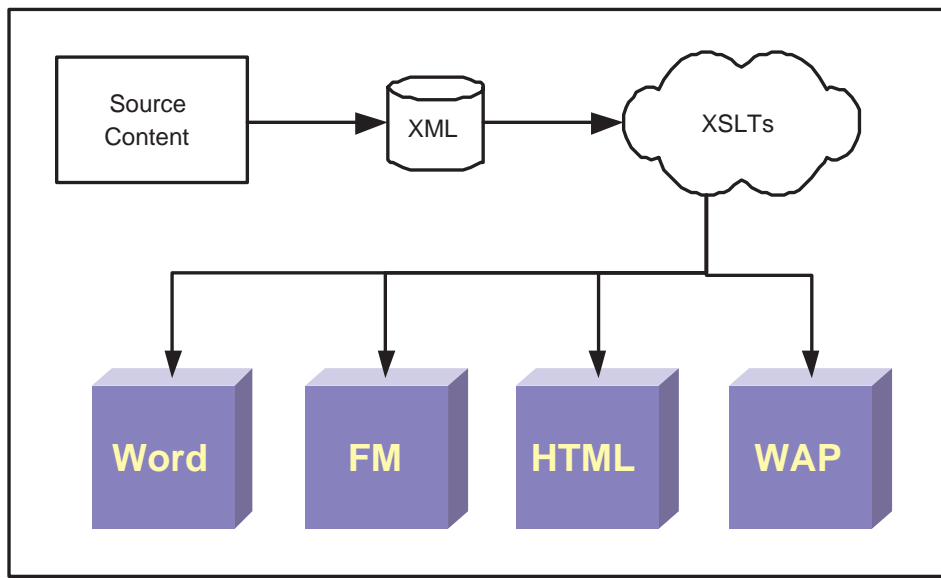
Because each of the content elements is appropriately structured, it fits together with other elements to form consistent document instances. In other words, each content object contains an introduction, main matter, illustrations (if any), conclusion and transitions. Several content objects are aggregated to form an document instance. Given a different use, it may be necessary to slightly modify some of the introductory or transitional materials. You also can structure your learning objects so that they can be specifically relevant to different user groups or to audiences of different aptitudes. The next document instance you require may contain the same content objects, which have been modified,

---

<sup>1</sup>the same people who brought your SGML and HTML

just a little here and there, to meet the requirements of the different audience.

The process of defining these structural components is analogous to creation of animation cells from layers. Each layer contains a different quality of information pertaining to the same object. These objects are then added together to produce an instance of learning delivery.



There are three main components of XML objects with which instructional designers need to be concerned: XML, DTD and XSLT. The XML file contains the content, just the content and only the content. The Document Type Definition (DTD) specifies how the XML is structured. The XML Style sheet (XSLT) contains all the formatting information for the output document instance.

## 2.1 Document Type Definitions (DTD)

A DTD in an XML document provides a list of the elements, attributes, comments, notes, and entities contained in the document. It also indicates their relationship to one another within the document. In other words, a DTD is the grammar of an XML document[10].

There are thousands of DTDs from which to choose. Most of them are related to a specific industry or enterprise. Unless you have unlimited time and capital, there probably is no reason to even contemplate making a custom DTD that models your core organizational elements. The most widely distributed DTD is DocBook[8], which originally was developed for documenting computer software. Many popular DTDs, such as the Telecom Interchange Markup (TIM) are derivatives of DocBook. Because XML is extensible, you can begin with a DTD that meets your core requirements and then make minor modifications and additions to it at a later date without invalidating all your XML.

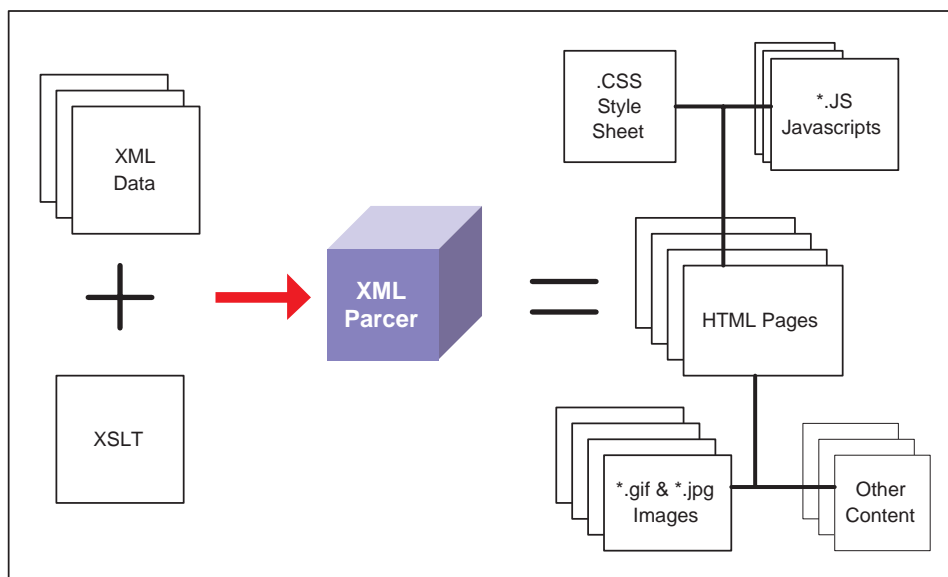
The DTD defines the taxonomy that you will use to parse information about the subject matter into re-usable form[2].

## 2.2 XSL and XSLT

XSL is the language used to create transformations (XSLT). XSLTs are the method used to transform raw XML data into useful content[11]. They are referred to as style sheets, but they are not style sheets in the same way as a cascading style sheet defines the look of an HTML document instance.

It is important to note that authoring in XML divorces the content from the format. The reason for this is quite simple: consistency. When you make a change to the source content of a series of deliverables, you should be able to produce those new versions with a minimum of intervention; ideally with no intervention. This is accomplished by creating an XSLT for each document instance type and then processing content through that common XSLT for all versions of that output deliverable type.

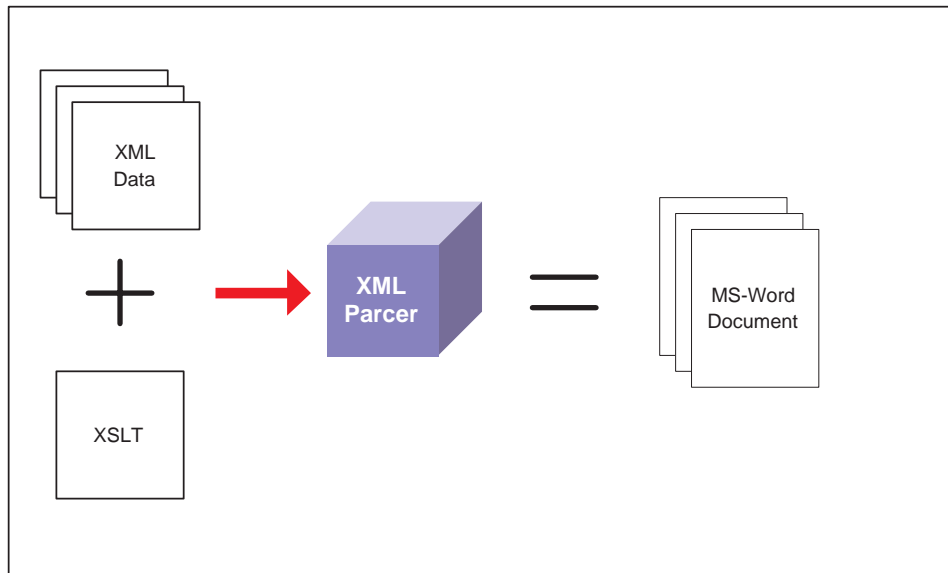
This is a departure from traditional word processing and publication tools, which merge these two operations into a single work flow. Some tools used to author XML, such as FrameMaker allow the designer the illusion of authoring of content with format, but all formatting is defined in the XSLT, which authors very seldom modify.



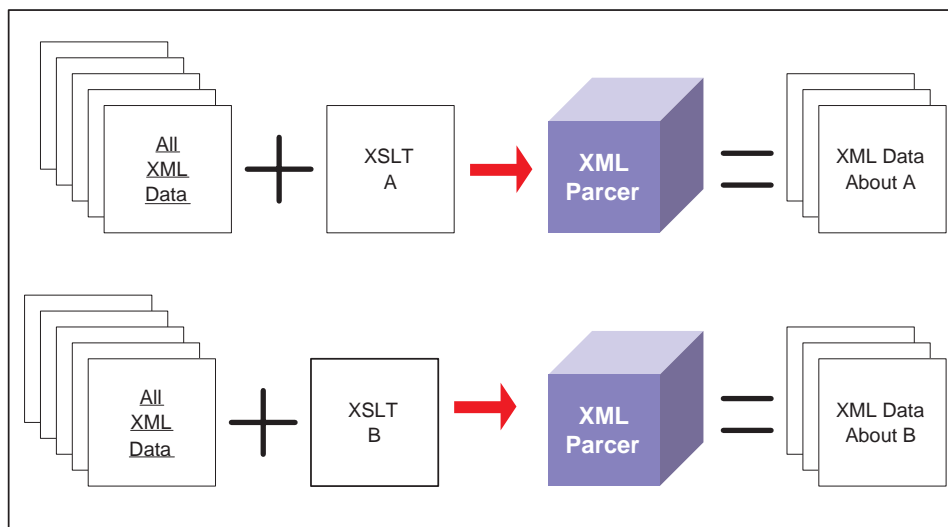
XML is data and contains no formatting. However, because it is structured data, the formatting can be applied to it programmatically. The XML data and the XSLT is parsed<sup>2</sup> to become a Web site. The Web site contains the HTML pages, images, javascripts, style sheets and other content associated with a full-featured Web site. The same content can be

<sup>2</sup>There are many kinds of parsing engines, come written in C++, Perl, Python and Java.

parsed with a different XSLT to result in WAP Web content or a Microsoft Word document.



An XSLT also can convert the same content to an Microsoft Word document (or any other document format that is desired).



Lastly, an XSLT can be used as an intelligent query to result in the XML content that conforms to a specific set of rules. These rules can be very complex and can be a more efficient way to query data than a direct Structured Query Language (SQL) query to the database.



## 2.3 What about ...?

The following sections describe several initiatives and protocols that are related to XML.

### 2.3.1 SCORM

To hear vendors talk, if your XML is not SCORM compliant, there is something wrong with you. SCORM is a set of standards, implemented in XML, which passes instructions to a Learning Management System (LMS). It is analogous to an application program interface (API). If your organization selects a SCORM compliant LMS, then your repository should contains an XSLT to present your content with additional SCORM elements that enable the LMS to know how to deal with it. The nice part about SCORM is that it is an open standard. If you decide that this LMS does not meet your needs, you will be able to select another SCORM-compliant LMS<sup>3</sup> and only have to tweak your repository here and there to make it all work correctly.

### 2.3.2 SOAP

SOAP[12] is a lightweight protocol for exchange of information in a decentralized, distributed environment. It is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined data types, and a convention for representing remote procedure calls and responses. Although usually associated with Microsoft, it is actually maintained by a consortium of companies, of which Microsoft is one.

### 2.3.3 .NET

.NET is a "software platform". It's a language-neutral environment for writing programs that easily and securely inter-operate. Rather than targeting a particular hardware/OS combination, programs will instead target ".NET", and will run wherever .NET is implemented. In this way, it is just like java. .NET is also the collective name given to various bits of software built upon the .NET platform. These are both products (Visual Studio.NET and Windows.NET Server, for instance) and services (such as Passport, Hail-Storm, and so on).

.NET uses XML in many instances, especially when complex data is transferred. It is a case of using what works best to accomplish some tasks. Other than this, it has very little

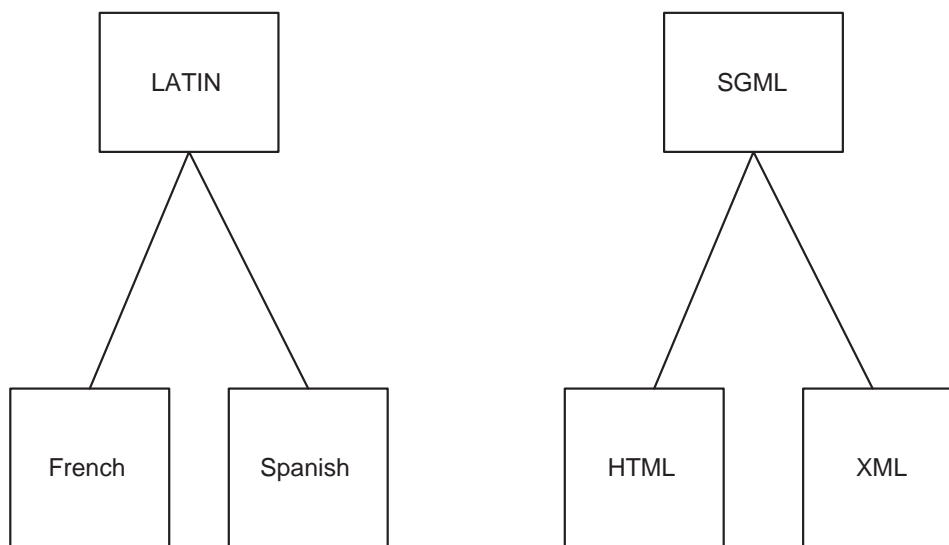
---

<sup>3</sup>Your mileage may vary. How SCORM-compliant is a piece of software? Unfortunately, it is often as compliant as the maker says it is, without any testing or third party verification.

to do with XML.

### 2.3.4 SGML

Standard Generalized Markup Language (SGML) has been around for a long time (1968) and is still still is a universal tool, both platform and system independent, for describing text. SGML is rather like Latin – the language from which other languages have been derived. Like Latin, it is more difficult to use in a contemporary context than its children. Also, there are many more tools being developed every day to do things in XML. This is not true of SGML anymore.



XML owes much of its functionality and usefulness to the lessons learned by people using SGML. SGML is too good, too good for humans to be able to use it well. Man does not live by macros alone; ergo XML.

## 3 Taxonomy

The application of a useful taxonomy to enterprise information is what determines whether the content reuse system achieves benefits for the organization or becomes just another expensive idea.

All learning objects are defined by taxonomies. These taxonomies express the way in which each object is understood, used and maintained. In evaluating how to construct learning object models for an XML repository, it is very important to understand that these are used to define queries. The value of the system depends upon the ease and accuracy of queries. Many organizations discovered too late that they had expended substantial resources in creating an XML (or SGML) repository that provided no additional benefit over cutting and pasting docs from a file server. This is because their content authors could not find anything that was placed into the repository.

### 3.1 Repository

The repository has several different purposes:

- Store controlled versions of documents
- Store current versions of learning objects
- Store in-work versions of learning objects
- Publish content to web servers
- Publish content to other servers (LMS)
- Function as ISO Repository

Once everyone has gone through the arduous task of chunking and labeling their legacy content, this content needs to be put into a repository where it can be easily accessed. The best way to do this, for instructional designers, is to put the content into a version control system that is linked to a database. ClearCase, for example, can present several different views of the repository for different uses. One view presents a virtual file server that contains all the most recent versions of the training documents. Another view presents selected documents to a web server or LMS. Yet another view presents the XML database elements.

Other views can be created for specific uses, such as creating archives of content, presenting catalogs of approved artwork or source content for other servers such as Adobe Document Server or FrameMaker Server.

The first task of the repository is to facilitate collaboration between content creators, editors and production staff. One mistake often made with a complex repository is to make

customized views that are not shared between different team members. This can be frustrating and time consuming.

The road to XML Content Reuse is a simple progression of responses faced by learning organizations. The following table summarizes some of the objectives and limitations of each step in the progression:

Step	Objective	Limitation
<b>File Server</b>	To permit access and sharing of files between many users	Slow, insecure and does not scale well
<b>Version Control System</b>	To maintain different versions of the same document so that the newest (best) can be identified	Complex to maintain and difficult to use when additional features are added
<b>Doc Manager</b>	To automate higher features	Proprietary - software does not keep pace with new tools and processes
<b>LMS</b>	To improve the efficiency of training content delivery and progress tracking	Limits IDs in terms of format or delivery methods.
<b>LCMS</b>	To improve the efficiency of training development through content management and reuse	Poor user interface; extensive customization required
<b>XML Repository</b>	To provide content reuse, multiple output formats, and extensibility to react to changing needs	Users resist new work methods and process

When computer networks became common in the workplace, people abandoned the file cabinet for the file server. They soon learned that file servers have their own defects when it comes to sharing important information. The next logical step was to try to remove the most glaring defects of the file server by implementing a version control system. The version control system made it easier to find things, but when you have enough things, it gets harder again. Enter the document manager, which makes it simpler to find things, but which usually required that you use outmoded tools and processes.

Learning Management Systems are student-facing applications, primarily. Although more and more content management facets have been sneaking into these learning delivery platforms, that is not their core function. The biggest drawback of the typical LCMS is that it locks you into tools and processes that compromise its own efficiency.

Once you have an XML repository, your repository can inter-operate with other systems, such as LMS or even LCMS, but the content is organized for your exclusive needs and convenience. If your needs or tools change, so can the repository. Why doesn't anyone market an XML repository as an LCMS? Because LCMS vendors sell you purposefully proprietary systems. The architecture of their product is intended to keep you from being able to use any part of the investment you made in the LCMS if you happen to part company with that vendor in the future. Unless you are intending to market the XML

repository you make for your organization, you have no reason to follow their example. For that reason, the XML repository is more direct and less difficult to upgrade.

XML and SGML were developed specifically to provide a structure and methodology for content reuse. Many of the lessons learned from early SGML implementations were built into XML, which provides a more streamlined and less labor-intensive means of achieving high quality content reuse.

Before legacy content is parsed into XML, it exists in many different forms. Most of these forms represent document instances. Most organizations attempt to maintain a repository of these document instances according to some meaningful hierarchy. ISO documentation standards are an example of this kind of document-centric hierarchy. If documents are correctly named, stored and updated, then the information they contain can be reused, but the process is slow, laborious and prone to failure. The utility of simple file sharing is inversely proportional to the number of documents to be shared.

When content is chunked, it is broken into topics and then broken again into smaller pieces identified as introduction, main body, and transitions. We like to have content sound natural and appear to have been written specifically for each use. Content also is chunked by complexity so that more complex material can be added or removed easily.

Audience also plays a big role in content reuse. Identifying specific blocks of information as appropriate or inappropriate for different audiences can simplify document creation immensely. It also is the hardest classification to accomplish.

For example, an Offer Brief: It is going to have the same look and feel, auxiliary content, and illustrations as 50 others, but the new one will be different in three main content blocks. The task is now to find the 4-5 other Offer Briefs that contain almost the same information. You could do a keyword or phrase search, but that would take a long time and produce indiscriminate results. With a properly constituted XML repository, the process is more like playing 20 Questions, only, in this case, it is more like 10 questions (or less).

The only taxonomy that should not be used to chunk content are attributes of the content that change rapidly over time, such as department, owner or other descriptor that is irrelevant to the learning objectives.

## 4 Process

This section describes the development process to implement the XML content reuse system. Each description includes a discussion of the costs and benefits associated with each process.

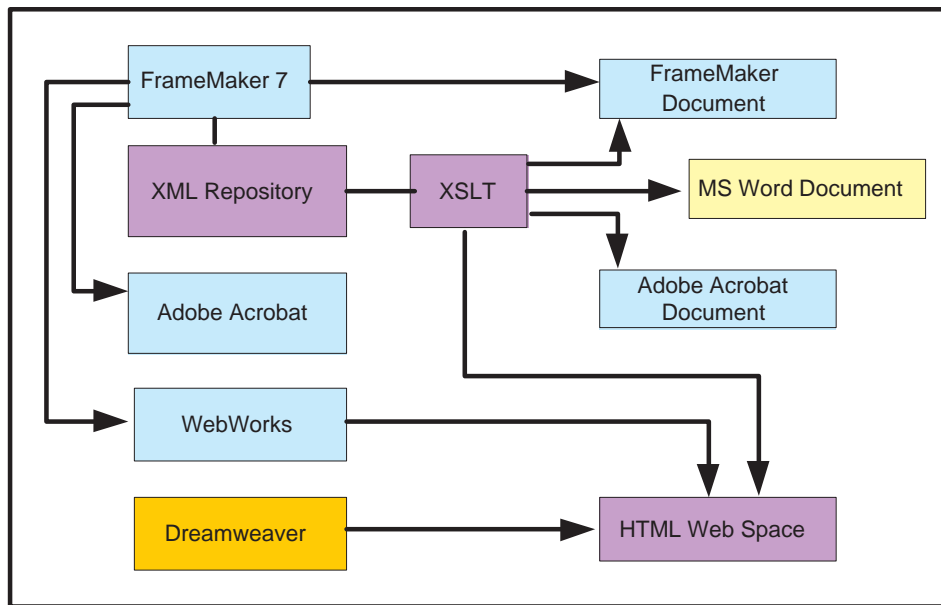
### 4.1 Ye Olde Way

In traditional, project-oriented design settings, each new project was a separate entity. Analysis, development, and production were defined by the time line and requirements of each discrete project, and instructional designers produced design and content as an artisan custom crafting a product for a customer. When this process has worked correctly, it has worked very well. Students receive curriculum that is specifically fashioned to address their needs. Trainers and designers can be student advocates at many different levels. Everybody wins.

This disadvantages of this methodology are many:

- **Inconsistency** - Since every project is independent of every other, it is very difficult to create and enforce standards. Even if templates are used, designers tend to create exceptions.
- **Inefficiency** - There are many opportunities for reusing content that are missed, either because designers are unaware of legacy content that could be adapted or because the legacy content is in a format that makes it difficult to adapt to their current project.
- **Inaccuracy** - Because each project recasts some of the same information in a different way, there is no way to globally update information and reissue training when changes occur.
- **Scalability** - As workloads increase and staffing levels decline, there is no way to maintain output and quality levels. Designers become frustrated, being unable to meet the expectations of their audience.
- **Tool Costs** - Reliance on outmoded tools, different versions of standard tools and fringe tools complicates things and makes people less efficient. The cost of maintaining learning materials sourced in multiple tools is enormous.

## 4.2 The XML Way



The structured approach to instructional design is seen to have the following benefits[6]:

- Allows the same courses to be delivered across multiple media and delivery environments.
- Supports a consistent instructional design and development process
- Provides a definitive view, including meta-data, of the components of well-constructed educational resources responsive to different learner profiles
- Provides opportunities for learners to approach the course material through multiple paths or views.
- Facilitates the re-purposing and updating of content.
- Conforms to Information Technology standards to ensure portability and long-term use.

There are three steps in the process of implementing an XML content reuse system: 1) Analysis, 2) Chunking, 3) Operation. The process is very simple, in theory:

- A DTD is selected and tested.
- The repository is created using tables that mirror the DTD.
- Legacy content is converted to XML.
- XML content is placed in the repository.
- Users query the database to construct new documents

- Users add new content to the repository as needed.

As mentioned before, the initial analysis is perhaps the most difficult stage of the implementation and it is the one stage that has the most persistent effects. Having once decided upon the one and only way the content will be parsed, staff members are trained carefully in how to accomplish the chunking of legacy content into the system.

### 4.3 Legacy Content Chunking

Whether this chunking process is slow and manual or quick and automated really depends on how much legacy content was created using standardized styles and content properly. If the answer is no, then there is a great deal of manual evaluation that must be done.

The most important aspect of the chunking process is thorough training and supervision of the people who will be doing this important work. Consistency is the key. Select a single process, train everyone in that process and execute the process without exception.

**NOTE:** The importance of thorough and consistent content editing increases by several orders of magnitude when content is entered into the database. *Enter it wrong once ... use it wrong many times.*

“Organizations that implement highly configurable or customizable products need to rely on their software vendors to meet the early training needs of the planners and technicians. To the degree that they wish to own or control product configuration, customization, and the ongoing support of those modifications, they also need to be prepared to invest in the staff development required to enable those capabilities.”[4]

There are two approaches to legacy content that are usually successful. The first identifies a small select team of designers who work on converting content and nothing else until the original body of required content has been put into the database. The other way is to spread the conversion among all the design team, with each member converting documents among their other duties, but at least x number of hours per week.

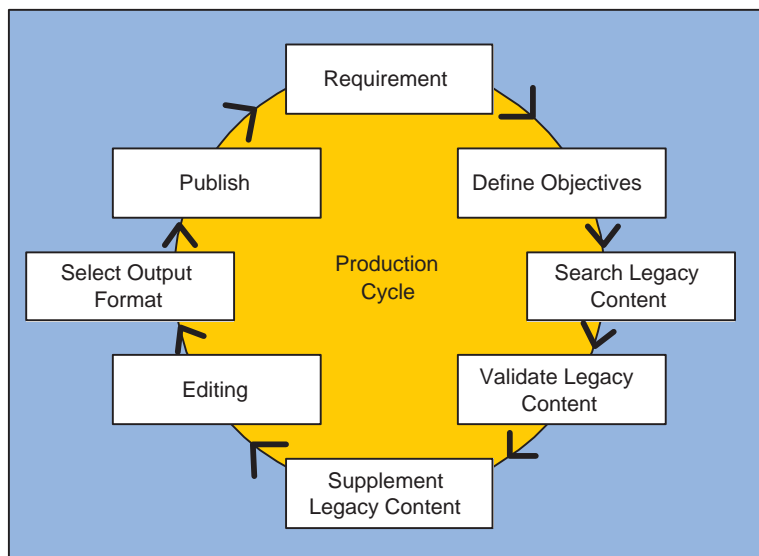
The advantage of the first method is that you generally obtain a more consistent conversion with fewer errors. The advantage of the second method is that you train your entire group in the XML database and process. You also may learn some things early on that allow you to modify the database or your processes so that they are more applicable to your training. Those who follow the first method may find themselves with a fully functional content base and no one trained to use it.



## 4.4 Using Chunked Content

The theory of developing new documents from legacy components is fairly simple, if the repository is implemented properly. First, the designer needs to know what previous training this new training is similar to. If it is completely new and dissimilar from other training, then the designer gets nothing from the repository but templates. Having made a shrewd guess about some other similar training, the designer has to define how this new training is different from the similar training that has been identified.

One method of handling the query process is by a web page containing drop down field list properties. Define 5 or 6 of these and then add in some more specific customizing terms, click submit and get a list back of matching content. It is just like doing a web search, except that the web you are searching is a discrete database. What is returned from the search can take many different forms: FrameMaker document, raw XML, Word document or HTML.



Some authoring environments, such as Epic Editor, work from the data structure to the content. At the beginning, these tools can be difficult for some designers to understand and use efficiently. After the designers become familiar with the database structure, they rapidly learn to navigate through the maze of information they encounter on cross-functional teams to find the parcels they want. In practice, authors working with common, standardized documents rapidly learn what five or six elements they must identify to generate the greater portion of their training. Even in a pure XML environment, designers still find invaluable the ability to easily query the database.

In practice, authors working with common, standardized documents rapidly learn what five or six element attributes they have to specify to see 90% of the pertinent content to their training. It is more difficult, at the beginning, than cutting and pasting content, but once you get it into your stride, it becomes ten times faster and easier to do your job.

How the content is organized into new instances is a question of authoring tools, not XML.

## 4.5 Single Sourcing

In an enterprise environment, there are many uses to which information is put. Some of those uses include documentation, training, knowledge-base applications and marketing. Traditionally, these disparate uses have all maintained their separate knowledge management environments. As a result, the information provided by these various sources is usually inconsistent. In the worst case, considerable misinformation results.

When these information sources are unified into a single repository, from which all outputs derive, significant improvements in efficiency, consistency and overall quality of information result. Also, when the costs of implementing the content repository are spread among different organizations within the enterprise, a greater return on investment naturally occurs.

Communication is the unstated core competency of every successful business. When the information about its products, processes, policies and procedures is available to all associates, this has a unifying effect on all the organizations within the enterprise. Although the process and deliverables of different organizations vary tremendously, their need for accurate and timely information is identical.

In its best form, the XML content repository can be a significant competitive advantage to an enterprise, particularly one that operates in diverse markets. In this sense, the economies and productivity conferred to the training organization are a byproduct of a larger benefit to the entire enterprise.

## 5 Return on Investment

According to Microsoft[7], if you have 100 employees using a system that cost you \$1,120,000 initially and requires a staff of 4 IT professionals to run it, the system must return an increase in productivity of 16% to break even the first year. Whether you spent this money on developing your own system or purchasing someone else's system, it is money well spent.

Collecting increased productivity from instructional designers depends on a number of factors:

- How much of the designer's time is spent actually developing content, as opposed to time spent developing curricula?
- How much of your content would actually get reused? In some organizations, the quantity might be almost nil and in others it could approach 50% or even more.
- How many repetitive training operations do your designers perform in order to get content to the students?
- Do your designers have the willingness and ability to change their methods?
- Does your organization possess the infrastructure and the commitment to design or customize, implement and use the content reuse system?

Assume that the instructional designers in your organization spend approximately 60% of their time accomplishing some aspect of content development. Further assume that of this time, roughly two-thirds would be generally unaffected by content reuse. If the content reuse system you implement results in that employee becoming 15% more productive in these tasks, then the net result for that employee is a 6% increase in productivity overall.

For every organization, there is some training that is mandatory, some that is essential, some that is preferred and some that is optional. Mandatory training represents those training hours that must be delivered to meet statutory or contractual obligations. Essential training provides to the employees the skills and knowledge they need to perform their jobs to a minimum standard. Optional training provides the employees with the skills and knowledge to excel. In today's tough economic conditions, many organizations have been trimming their training efforts to such an extent that they are beginning to see negative productivity results.

Faced with such realities, these organizations are faced with the challenge of providing a competitive, sustainable solution to obtaining quality training that facilitates excellence. An XML based code reuse system qualifies as an excellent example of such a solution.

## 6 Conclusion

Implementing a robust content management system for the enterprise is not just an idea whose time has come, but an idea long overdue. Far too often, training departments must make hard decisions in hard times that end up being false economies. The underlying technology of XML has been proven in numerous settings over the past twenty years, first in government and then in the private sector. It is rapidly getting to be the case that if your enterprise does not implement such a system, you will compete at a considerable disadvantage in your marketplace.

In summary, the advantages conferred by the XML content repository and reuse system are:

- **Economy** - Provides the same economies of scale as automating any labor intensive, customized process.
- **Communication** - Provides a naturally unifying influence on the organizational communication; both internally and customer-facing.
- **Quality** - Provides the ability to attain higher quality levels and increased consistency across all training and similarly-sourced deliverables.
- **Productivity** - Provides additional capacity to produce mandatory training at a lower cost. Confers surplus capacity that can be used to develop more effective training that raises enterprise-wide productivity.

## References

- [1] Robert Eckstein With Michel Casabianca. *XML Pocket Reference*. O'Reilly, Sebastopol, CA, 2001.
- [2] W. Scott Means Elliotte Rusty Harold. *XML in a Nutshell*. O'Reilly, Sebastopol, CA, 2002.
- [3] Willaim Horton and Katherine Horton. *E-Learning Tools and Technologies*. Wiley, Indianapolis, IN, 2003.
- [4] Michael Hughes. Keeping just-in-time from being way-too-little. *Performance Improvement*, 42(6):37–40, July 2003.
- [5] Jan Kampherbeek. *Crash Course in XML*. <http://www.spiderpro.com/bu/buxmlm001.html>, June, 2001.
- [6] Gerry Paille Solvig Norman Prescott Klassen John Maxwell. *The Effect of Using Structured Documents (SGML) in Instructional Design*. <http://naweb.unb.ca/proceedings/1999/paille/paille.html>, February, 1999.
- [7] Microsoft. *Content Management Server, Return On Investment Calculator*. <http://www.microsoft.com/cmserver/evaluation/roicalculator.xls>, 2003.
- [8] Leonard Mueller Norman Walsh. *Docbook, the Definitive Guide*. O'Reilly, Sebastopol, CA, 1999.
- [9] George M. Piskurich. *Rapid Instructional Design*. Jossey-Bass/Pfeiffer (Wily), San Francisco, CA, 2000.
- [10] Erik T. Ray. *Learning XML*. O'Reilly, Sebastopol, CA, 2001.
- [11] Doug Tidwell. *XSLT*. O'Reilly, Sebastopol, CA, 2001.
- [12] W3C. *Simple Object Access Protocol (SOAP) 1.1, W3C Note*. <http://www.w3.org/TR/SOAP/>, May, 2000.

# Index

abstract, 1

chunking, 13

chunking, legacy content, 16

chunking, reuse, 17

content element, 5

docbook, 6

dtd, 6

introduction, 2

lcms, 12

learning object, 2

lms, 12

net, .net, 9

process, 14

process, traditional, 14

process, xml method, 15

repository, 11

roi, 19

scorm, 9

sgml, 13

smart tags, 5

soap, 9

SQL, 8

taxonomy, 11

terminology, 3

wap, 8

xml definition, 5

xml parcer, 7

xml perspective, 2

xsl, 7

xslt, 7