XML for Information Management Workshop Talking Points

ishlist points in the script and slides where you may want to customize the workshop to include case studies and examples from your own experiences, and exercises which may be more appropriate for your audiences.

Customization Index
(listed by slide number)

Workshop background, Acknowledgements, and Contact Information

Introduction (Average 25 minutes, 15 minute participant introductions (this unit takes less time during the XML workshop, because it is often presented the day after the Metadata for Information Resources workshop))
Intro-3
Intro-4

Unit I (Average 1 hour 8 minutes, includes 20 minute joke exercise)  Unit IV (Average 11 minutes)
I-2  IV-3
I-3  IV-4
I-5  IV-5
I-6
I-13
I-14
I-15
I-16
I-17
I-18
I-19
I-20
I-21

Unit II (Average 48 minutes)
II-2  VIII-12
II-5  VIII-14
II-8
II-9

Unit III (Average 24 minutes)
III-4
III-5
### Workshop Background, Acknowledgements, and Contact Information

**Ack-1**

This workshop and all related materials are the direct result of a two-year grant to the State Archives Department of the Minnesota Historical Society (MHS) from the National Historical Publications and Records Commission (NHPRC). Work on the “Educating Archivists and Their Constituencies” project began in January 2001 and was completed in May 2003.

The project sought to address a critical responsibility that archives have discovered in their work with electronic records: the persistent need to educate a variety of constituencies about the principles, products, and resources necessary to implement archival considerations in the application of information technology to government functions. Several other goals were also supported:

- raising the level of knowledge and understanding of essential electronic records skills and tools among archivists,
- helping archivists reach the electronic records creators who are their key constituencies,
- providing the means to form with those constituencies communities of learning that will support and sustain collaboration, and
- raising the profile of archivists in their own organizations an promoting their involvement in the design and analysis of recordkeeping systems.

MHS administered the project and worked in collaboration with several partners: the Delaware Public Archives, the Indiana University Archives, the Ohio Historical Society, the San Diego Supercomputer Center, the Smithsonian Institution

---

**Note to instructor:** Include your contact information at the bottom of the slide
Archives, and the State of Kentucky. This list represents a variety of institutions, records environments, constituencies, needs, and levels of electronic records expertise. At MHS, Robert Horton served as the Project Director, Shawn Rounds as the Project Manager, and Jennifer Johnson as the Project Archivist.

MHS gratefully acknowledges the contribution of Advanced Strategies, Inc. (ASI) of Atlanta, Georgia, and Saint Paul, Minnesota, which specializes in a user-centric approach to all aspects of information technology planning and implementation. MHS project staff received training and guidance from ASI in adult education strategies and workshop development. The format of this course book is directly based on the design used by ASI in its own classes. For more information about ASI, visit http://www.advstr.com/

For more information regarding the workshop, contact MHS staff or visit the workshop web site at http://www.mnhs.org/preserve/records/workshops/edarchivists.html

---

### Introduction

**Introduction {Average 25 minutes, 15 minute participant introductions (if this workshop is presented the day after the Metadata for Information Resources workshop, this unit will probably take less time)}**

Note to instructor: It is helpful to handout all the exercises and examples at the beginning of the day, when handing out course books, so that you do not have to interrupt the workshop later. Consider copying them on sheets of different colored paper so that they are easy to distinguish from one another.

<table>
<thead>
<tr>
<th>Intro-1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>This unit includes:</strong></td>
</tr>
<tr>
<td>• Course objectives</td>
</tr>
<tr>
<td>• Course approach</td>
</tr>
<tr>
<td>• Course schedule</td>
</tr>
<tr>
<td>• Administrative items</td>
</tr>
<tr>
<td>• Participant introductions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Intro-1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>During the introduction</strong></td>
</tr>
<tr>
<td>• give an idea of what to expect throughout the day</td>
</tr>
<tr>
<td>• cover the course schedule</td>
</tr>
<tr>
<td>• administrative items</td>
</tr>
<tr>
<td>• introduce yourselves.</td>
</tr>
</tbody>
</table>
### Intro-2
#### Course objectives

Upon completion of this course, you will be able to:
- understand basic information technology concepts and terminology
- understand what XML is and why it is useful
- understand the reasons for the development of XML
- comprehend the relationship between metadata and XML
- recognize XML markup
- identify other components of the XML standard
- identify XML tools and editors
- understand how XML is being implemented in various projects

### Intro-2
Discuss the list shown on the slide.

---

### Intro-3
#### Course approach

- **Spiral approach**
- **About the exercises**
  - The exercises in this course will be “live”
  - The exercises will be limited in number
  - The exercises may require some oral participation from students
  - There is not a 100% correct answer; it is impossible to get an A, because there are many “right” answers
  - Today our exercises will focus on a recipe
    - Recipes are familiar to everyone
    - There are various things you are able to do with a recipe, which fit nicely with the variety of things you are able to do with XML
    - Recipes have obvious, tangible results
  - Parking lot
  - Team teaching
- **The spiral approach of teaching**
  - Introduce concepts and then revisit them several times throughout the day, each time expanding knowledge and pulling in more information, yet reviewing what has already been covered and showing how the concepts fit into the larger picture.
  - Begin by laying the foundation for the entire story and will elaborate upon it to greater levels of detail.
  - Spiraling is not just repeating, it is expanding and learning more each time you cycle through the spiral.
  - Because of the spiral, it may seem that we’re starting very slowly, and taking a while to actually begin talking about XML. And, in fact, we are, but there is a case (presenting XML in a larger context) to be made before we even get into the details of the topic, and using the spiral forces us to address that first.

There will be exercises throughout today’s class.
- will all be completed today during class,
- do not require homework,
- will not stretch over the lunch hour.
- will not be very many of them, but they do require participation.
- most of the exercises today will focus on a recipe and we will use the recipe to illustrate XML, showing it in various forms.
- recipe is something familiar to everyone, there are obvious tangible results when you think of a recipe, and there are various things you can do with a recipe, which fit nicely with the various things you can do with XML.

Yes, choose an example more appropriate for your audience.

- There are no “right” answers
The concept of a parking lot.

- Feel free to ask questions, but if I feel that we address your question better in a later unit, or if it’s out of the scope of the material we cover today, I’m going to put it in our parking lot, which is this posted piece of paper.
- Will either revisit the parking lot issue later, meet with you specifically during one of our breaks, or find an answer for you after this workshop and will contact you.

Both [name here] and I will be teaching at some point throughout the day.

- When one of us is teaching, the other will be acting as a coach, taking notes, noting questions, watching the time, and taking care of administrative items so the other person can continue to teach.
Here are the units we will be covering.

- **Unit I:** discover the definition of electronic records we will be working with today, and define what we mean by XML.
- **Unit II:** take a look at XML and become familiar with the markup.
- **Unit III:** look at the various XML standards available to present XML documents, primarily in a web format.
- **Unit IV:** cover free XML tools and editors and how you use them.
- **Unit V:** explore the growing family of XML standards.
- **Unit VI:** examine non-archival applications in using and applying XML.
- **Unit VII:** examine a specific archival initiative using XML: Encoded Archival Description, better known as EAD.
- **Unit VIII:** tie XML into use within an organization and some of the choices you will have to make.
- **Appendix A:** a list of the acronyms you’ll find throughout the course book.
- **Appendix B:** a list of XML tools and editors that are not free.
- **Appendix C:** a bibliography, a complete list of the citations and URLs in the workshop.
- Conclude for the day.

Note to instructor: You may wish to add an appendix offering pointers to local and national XML educators appropriate for your audience.

Schedule is very flexible
- Driven by your needs.
- Add more breaks if needed.
- If you have questions, or want to spend more time discussing a certain topic or unit, we will do so.

Discuss the items listed on the slide.

Class will begin promptly at the scheduled time.
- Need a break: squirm, lie down, tell me!
- If something is not clear, if you have a question, or if I say something that does not make any sense: stop me!
- Discussion is always better than lecture. Success of this class is dependent on all participants.
- Facilities (Parking, Restroom, Telephones, Exits)
- We will proceed informally.
### Participant introductions

- What experience have you had using XML?
- What projects are you participating in that currently use XML, or do you know of any projects using XML?
- Who do you know that is using XML?
- Who might you partner with for future XML projects?

---

### Unit 1: Defining XML

#### I-1

This unit includes:
- What do we mean by information resources, digital objects, and electronic records?
- Defining digital objects
- One person or organization cannot do it all.
- What is eXtensible Markup Language (XML)?
- Why XML?
- A quick definition of XML
- Explaining the XML standard.
- Comparing XML to Standard Generalized Markup Language and Hypertext Markup Language.
- Defining well-formed and valid XML

---

**Participant Introductions, Please take five minutes to fill out the following information about yourself, and then we’ll go around the room and share it.**

*After everyone has introduced themselves to the group, ask if there are any final questions before moving on.*
### I-2

What do we mean by information resources, digital objects, and electronic records?

**Information resources:** The content of your information technology projects (data, information, records, images, digital objects, etc.)

**Digital object:** Information that is inscribed on a tangible medium or that is stored in an electronic or other medium and is retrievable in perceivable form. An object created, generated, sent, communicated, received, or stored by electronic means.

An electronic record is a specific type of digital object with unique characteristics described by archivists and records managers.

**Types of digital objects:**

- e-mail
- Portable Document Format (PDF) files
- web pages
- PowerPoint presentations
- databases
- digital images
- spreadsheets
- word processing documents
  …and many more

### I-2

As we begin to discuss XML, let’s make sure we’re all on the same page by defining some of the terms we’ll be using today.

We need to start with a common definition of information resources, digital objects, and electronic records in order to understand how we can use XML to manage them.

Define **Information resources** as: The content of your information technology projects (data, information, records, images, digital objects, etc.)

**Our definition of digital objects comes from E-Sign**

- The Electronic Signatures in Global and National Commerce Act passed by Congress in 1999 to create a common legal framework for electronic commerce and electronic government in the nation.
- Digital object is defined as “Information that is inscribed on a tangible medium or that is stored in an electronic or other medium and is retrievable in perceivable form. An object created, generated, sent, communicated, received, or stored by electronic means.”
- Electronic record
  - is a specific type of digital object with unique characteristics described by archivists and records managers.
  - very broad and generic definition, which is exactly why it was chosen
  - it’s not exclusive to anything – records, digital objects, data, information, knowledge, all words we may use interchangeably.

☑️ **Note to instructor:** You may want to add your own definitions here as they relate.

This is a good place to start.

- In a practical sense, we need to break that definition down right away - it’s too broad
- To determine what we want to manage and how, we need to be much more precise.
- We also need to consider that people often think in terms of types or genres, like e-mail, web pages, databases, word processing documents, and the like.
  - But it’s not enough to know what application or file format a digital object is linked to if we want them to be accessible for however long we may need them, especially if we need to share them, re-use them, and/or if they are expected to outlast their original systems

### I-3

**Digital objects have three components:**

- Archivists and records managers came up with a definition that applies to electronic objects, but we think it applies to
Content: Informational substance of the object. 
Structure: Technical characteristics of the objects (e.g., presentation, appearance, display). 
Context: Information outside the object which provides illumination or understanding about it, or assigns meaning to it.

all information objects, and it is applicable to anything in a digital format. Content, Structure, and Context was first defined by the Pittsburgh Project in the early 1990s which focused on helping archivists deal with information Technology and electronic records. It’s just one of the many ways to describe a record or an object.

- Content: Informational substance of the object. (What it says)
- Structure: Technical characteristics of the objects (e.g.; presentation, appearance, display). (How the record looks)
- Context: Information outside the object which provides illumination or understanding about it, or assigns meaning to it. (What it is about)

Illustration 1: check example: [Draw a diagram of a check or have an actual one to hold up]
- Content: name, amount
- Context: produced by a bank (authority)
- Structure: where things are placed, if out of order will not be accepted as a valid check

Illustration 2: $20 bill example [hold one up]
- Content: basic information about the bill - $20, serial number, image (2 Jacksons), statement about legal tender. information substance of records
- Structure: what says that this object is authentic - hologram/ghost image, hidden strip, color of the ink, feel of the paper, etc. The structure assures us that it is valid, we doubt it it’s validity if any of these components is missing.
- Context: information outside the object. foreign currency market – how related to other currency, what’s it worth
  - What if you saw a photocopy of a dollar bill? Why wouldn’t that be an authentic object? It would be missing the necessary components

Content, Context, and Structure are the necessary components to help us understand what an object is and what it’s worth

I-4 Defining information objects

<table>
<thead>
<tr>
<th>Pittsburgh</th>
<th>Order of Values</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content</td>
<td>Data</td>
<td>Data</td>
</tr>
<tr>
<td>Structure</td>
<td>Information</td>
<td>Format</td>
</tr>
<tr>
<td>Context</td>
<td>Knowledge</td>
<td>Application</td>
</tr>
</tbody>
</table>

I-4 Take those three components and compare it to some other ways to define digital objects.

In the first column we have the scheme we just talked about.

Move to the second column and read down. Many people break down information objects in terms of the value they represent to an organization.

- one may say that it’s data, information or knowledge.
- look at this column in terms of our $20 bill again.
<table>
<thead>
<tr>
<th>Context</th>
<th>Knowledge</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data here is the lowest common denominator. $20 is the equivalent of 2000 pennies, but if you wanted to purchase something for $14.97, would you be welcome anywhere if you pulled out a bag of pennies and said, “Just wait, I have exact change!” Data is accurate, but it’s not necessarily useful. It’s the least functional value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Information – here data is structured for more functionality. It’s data presented in a practical format. A $20 bill is more useful than 2000 pennies. It’s in a specific format that’s designed to be more easy to use.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Knowledge – is data available for a wide variety of uses. Think of $20 in the bank, with a variety of uses and means. You have different ways to access it through a check or debit card. You can automatically withdraw money to pay bills. It can earn interest or the bank can use it to loan to other people. This is the level where you get the most value.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Why isn’t this enough? Let’s look at the third column.

- To use technology, all of these components and values have to be captured in a specific information technology architecture, a configuration of hardware and software that allows us to use computers to manage our information.
- And an architecture has three components:
  - data – the actual stored bits and bytes
  - format – data in a particular format (e.g., Word file, PDF, TIFF image) (how it looks to us)
  - application – a program that takes a particular format and puts the data to use, gives it some functionality.
- The architecture traditionally represents a limitation.
- Applications are subject to rapid obsolescence. And, they often don’t do all the things we want them to do.
  - we try to take data configured for one application and make it work in another. We lose a lot in the process usually. Because applications and their associated formats are usually proprietary something is almost always lost in the process. Very often, we’re lucky to preserve just the data, let alone structure and functionality, given our limitations.
- The $64,000 question for archivists and for anyone trying to preserve an investment in information. How do we preserve the value of our asset, our knowledge, over time, when the tools that help us realize that value (the hardware and software) are so unreliable?
That's where metadata and XML come in.

- Metadata allows us to capture the pertinent information, the content, context, and structure.
- XML allows us to preserve the information, to wrap records and make them useful across systems. Allows for knowledge and information in an executable format. And the format allows us to take advantage of the metadata.
- The key is that XML is infrastructure independent. You can wrap up your knowledge and apply it in different configurations of hardware and software, protect it from obsolescence and maintain its value over time. It's executable knowledge because we've captured the knowledge with the ability to use it within an architecture that is not tied to specific hardware or software.
- XML might not always be the answer for you - that's one thing we will talk about. Something else will undoubtedly come along. Some other choice might be more appropriate or cost-effective for you right now. So we want to discuss when and why XML is the right choice. But it is undoubtedly a viable and effective choice. Even if there's a revolution 5 years from now and YML comes along, you'll be much better placed to take advantage of it, if you're using XML now.
- This workshop is designed to give you the information that will help you make appropriate decisions about XML.

<table>
<thead>
<tr>
<th>One person/organization cannot do it all</th>
<th>Given the sheer amount of information and digital objects created by any one person or agency/organization, it's obvious that one person and one organization cannot do it all or do it alone.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Components: X + Y + Z = resources, tools, standards/methodologies, education, technology, partners</td>
<td>One way to tackle digital objects is through a sustainable plan. First you need to decide at the beginning if XML is right for you.</td>
</tr>
<tr>
<td>Resources: skilled and informed staff, organizational funding, grants, …</td>
<td>In order to bridge XML from a concept, you're going to need to be able to implement it as a program. Technology within a context. To do this you need a sustainable plan. Something which moves you from the conceptual to the actual, and sets you up for long-term success. This is an outline of the variables you will have to address in getting you from here to there.</td>
</tr>
<tr>
<td>Tools: metadata, markup languages, …</td>
<td>Every organization will place varying levels of importance on the following components or variables, but in general you need to have all of the following, to some degree, in order to have a successful sustainable program.</td>
</tr>
<tr>
<td>Standards/methodologies: Dublin Core, XML, …</td>
<td>You will need to ask yourself, what is appropriate for you and your organization.</td>
</tr>
<tr>
<td>Education: formal/informal education, workshops, conferences, …</td>
<td>Start to fill in the variables, which are all dependent</td>
</tr>
<tr>
<td>Technology: TagGen, XMetal, …</td>
<td></td>
</tr>
</tbody>
</table>
This class will cover one tool and standard:
XML

upon your environment and your needs – what’s applicable for you:
  o resources, tools, standards/methodologies, education, technology, and partners
  • think of it in terms of an equation:

☑ Customize the example for your audience.
  o resources = skilled and informed staff, organizational funding, grants;
  o tools = metadata, markup languages;
  o standards/methodologies = Dublin Core, XML)
  o education = formal/informal education, workshops, conferences;
  o technology = TagGen; XMetal
  o partners = stakeholders [record creators, agencies, users], other archivists, professional organizations [Midwest Archives Conference (MAC), Society of American Archivists (SAA), National Association of Government Archives and Records Administrators (NAGARA), Association of Records Managers and Administrators (ARMA)

The plan is very scalable.
  • All archivists will have to make decisions about these components and how they fit into their program.

This class contributes to two of the components:
  • Education: Because you and your employer value education as a component of your sustainable information management program, you are here today.
  • XML: A tool and standard. It is a standard, how do we make it a tool? It’s not a hammer that you can just pick up and use, you have to apply it in certain and specific ways to get it to do what you want. There is an investment on your part, and a learning curve. But, that’s why you’re here today.

I-6
Exercise: What do you think eXtensible Markup Language is?

I-6
Exercise: What do you think XML is?

A brainstorming session,
  • Feel free to throw out ideas.
  • Don’t expect an ideal definition or perfect understanding right now, because we’re going to explain XML to you throughout the day.
  • We’ll revisit this list throughout the day to understand what XML actually is and is capable of and how that fits with the preconceptions we identify here.

☑ Note to instructor: Use a flip chart to write down responses. Post the sheets so that you can refer back to them throughout the day.
I-7
Language means communication and communication leads to understanding

What makes understanding possible?
- vocabulary
- dictionary
- grammar

It’s not just semantics. This is the structure of an “unstructured” text.

I-7
Let’s focus in one what you’ve said about XML, particularly the language and communication aspects of it. Today we’re going to explore XML as part of a language. Why do we want to do this? Language means communication and communication, ideally, leads to understanding.

What makes understanding possible? When we're talking about a language, it's the following items (think of learning a second language):
- vocabulary - words
- dictionary – definitions, rules of pronunciation, words you string together according to a grammar (proper meaning)
- grammar – how to form sentences and create meaning and communicate with them in an accepted way

In order to have some common understanding, we need consensus about what we’re saying. Need to put all of our ideas together in an understandable format.

To create meaning…
- It’s not just semantics. This is the structure of an “unstructured” text. Turns into executable knowledge.

Quotation: “When I say a word, it means exactly what I want it to mean.” Humpty Dumpty – Alice in Wonderland
- Humpty Dumpty is one way to look at it.
- He didn’t care if other people understood him.
- He illustrates the good an the bad about XML:
  - Yes, we have the potential to create our own language
  - But, if no one understands it and it only makes sense to you, what’s the point?
  - With XML we want to create a language we have control over and that we will use to communicate with, otherwise you’re just talking to yourself

The implication is XML has to mean something to everyone who needs to understand it, and building a commonly understood language is a part of this.

I-8
What does eXtensible Markup Language mean?
eXtensible: In XML, you create the tags you want to use. XML extends your ability to describe a document, letting you define meaningful tags for your applications. For example, if your document contains many glossary terms, you can create a tag called <glossary> for those terms. If it contains employee identification numbers, you could use an <employeeid> tag. You can create as few or as many tags as you need.

I-8
So how do we work what we’ve just discussed with language into a standard definition? Let’s explore what the terms eXtensible Markup Language mean
- eXtensible: You get to assign value to the text. Extensibility means independence. While there are a few simple rules to follow, you are free to assign meaning to your documents as you see fit. You determine what’s important about your document.
  - This is a little disconcerting at first because there is no reference guide to look up what tags to use. But it also gives you great freedom and flexibility because you can define and use tags in a way that makes sense for your documents.
**Markup**: Any means of making explicit an interpretation of a text. In this instance, a notation for writing text with tags. The tags may indicate the structure of the text, they may have names and attributes, and they enclose a part of the text.

**Language**: XML is designed to facilitate communication. It follows a firm set of rules that allow you to say what you want in a way that others will understand. It may let you create an extensible set of markup tags, but its structure and syntax remain firm and clearly defined.

- Extensibility means you get more options and more power - but with those capabilities comes a need for planning. You will need to plan what tags to create. (You have to use that power for good)

- **Markup**: Structure of text within tags. Markup helps you keep track of that meaning. It’s structure applied to unstructured data that helps you to capture the value of your information and documents. Make explicit your interpretation of a particular text.

- **Language**: Based on a set of rules – define your values and markup in ways that others understand. Remember the goal of XML is communication. Language is what you use when you want to talk to someone, or tell people about your information.

**eXtensible Markup Language** gives you flexibility and creativity, all within a dictionary and grammar that have been defined so that everyone understands the language you are using.
Why XML?

- Share data: Different organizations rarely use the same tools to create and read data. XML can be used to store any kind of structured information, and to enclose or encapsulate it in order to pass the information between different computing systems which would otherwise be unable to communicate.

- Reuse data: XML documents can be moved to any format on any platform - without the elements losing their meaning. This means you can publish the same information to a web browser, or a personal digital assistant (PDA), and each device would use the information appropriately. XML can be designed in such a way that fragments or chunks can be pulled out of any given context and reused. So, when a chunk is updated, the resources that use the chunk are updated also.

- Customize data: XML allows for the development of user-defined document types. Users define the XML tags they want to encapsulate their data. XML also allows groups of people or organizations to create their own customized markup languages for exchanging information in their domain.

So, why do people want to use XML? Three reasons come to mind: to share, reuse, and customize data.

- Share data: We've all made significant investments of time, money, and resources in our technological environments and in maintaining our data. So, how do we break down our silos of information to share data when we want to and when we have to? How do we share data without changing our legacy systems?
  - Because different organizations (or even different parts of the same organization) rarely standardize on a single set of tools, it takes a significant amount of work for two groups to communicate. XML makes it easy to send structured data across the web/or network so that nothing gets lost in translation. When using XML, I can receive XML-tagged data from your system, and you can receive XML-tagged data from mine. Neither of us has to know how the other's system is organized. If another partner or supplier teams up with my organization, I don't have to write code to exchange data with their system. I simply require them to follow the document rules defined in the Document Type Definition, which describes how my XML documents work.

- Reuse data: Technical obsolescence occurs at an amazing rate, yet we want the value of our data to remain consistent and constant. How do we best translate this data, and migrate it from system to system? Also, how do we make data available now on all the different systems we currently use: computer, phone, PDA? XML helps us to structure information so it's useful on different devices.

- Customize data: We want to use data in a particular way, and we want to use it, enhance it, and package it with other data in common applications. XML allows us to get the most from our information resources that we can.

XML is an infrastructure-independent markup.
- Used to mark up records so they can be used by different systems and not dictated by particular hardware and software configurations.
- Not tied to one particular application, which is why it’s so valuable for breaking down those silos of information and allowing for the long-term preservation of records, because it allows for it’s use by a variety of different applications.

A Quick definition of XML

"Extensible Markup Language (XML): An initiative from the W3C defining an "extremely simple" dialect of SGML"
suitable for use on the World-Wide Web."

“XML: A metalanguage written in SGML that allows one to design a markup language, used to allow for the easy interchange of documents on the World Wide Web.” (Dictionary.com)

“XML (Extensible Markup Language) is a flexible way to create common information formats and share both the format and the data on the World Wide Web, intranets, and elsewhere.” (Whatis.com)

“The Extensible Markup Language (XML) is the universal format for structured documents and data on the Web.” (World Wide Web Consortium (W3C))

| suitable for use on the World-Wide Web. “XML: A metalanguage written in SGML that allows one to design a markup language, used to allow for the easy interchange of documents on the World Wide Web.” (Dictionary.com) | to communicate, a universal format.  
Not all of these definitions are the same.  
Why not? |
|---|---|


I-11
The XML standard

Ten Design Goals for XML:

1. XML shall be straightforwardly usable over the Internet.
2. XML shall support a wide variety of applications.
3. XML shall be compatible with SGML.
4. It shall be easy to write programs which process XML documents.
5. The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
6. XML documents should be human-legible and reasonably clear.
7. The XML design should be prepared quickly.
8. The design of XML shall be formal and concise.
9. XML documents shall be easy to create.
10. Terseness in XML markup is of minimal importance.

In other words, XML is easy to create, easy to read, and designed for use over the Internet.

The initiative to create XML began with the W3C the World Wide Web Consortium.

XML was developed by an XML Working Group (originally known as the SGML Editorial Review Board)
- formed in 1996.
- chaired by Jon Bosak of Sun Microsystems with the active participation of an XML Special Interest Group (previously known as the SGML Working Group) also organized by the W3C.
- SGML can be considered the parent of XML.
- The working group released the XML 1.0 recommendation in February 1998, which was revised in October 2000.

The standard
- describes terminology
- what is allowed in well-formed/valid XML markup
- what makes up an XML document
- abstracts how to process XML
- gives references to various resources

Since 1998 there have been other working groups addressing various parts of the XML specification such as
- namespaces,
- style sheet linking,
- XML query

The 1996 workshop group established 10 design goals for XML. Let’s take a quick look at these and whether or not their goals were met:

1. XML shall be straightforwardly usable over the Internet.
   - It is readable by certain browsers
2. XML shall support a wide variety of applications.
   - Yes, as evidenced by its use in a variety of communities (legal, real estate, meat and poultry) – we’ll look at in Unit VI
   - XML is meant to be license-free, platform-independent and well-supported. By choosing XML as the basis for a project, you gain access to a large and growing community of tools and engineers experienced in the technology.
   - Opting for XML is a bit like choosing SQL for databases: you still have to build your own database and your own programs and procedures that manipulate it, and there are many tools available and many people who can help you. And since XML is license-free, you can build your own software around it without paying anybody anything. The large and growing support means that you are also not tied to a single vendor.
3. XML shall be compatible with SGML.
   - It’s essentially a stripped down version of SGML.
1. XML is new, but not that new. Development of XML started in 1996 and has been a W3C Recommendation since February 1998. Before XML there was SGML, developed in the early 1980s, an ISO standard since 1986, and widely used for large documentation projects. The development of HTML started in 1990. The designers of XML simply took the best parts of SGML, guided by the experience with HTML, and produced something that is no less powerful than SGML, and vastly more regular and simple to use.

2. XML looks a bit like HTML. Which we'll see in a couple of slides

3. It shall be easy to write programs which process XML documents.
   - Good question, opinions from programmers???

4. The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
   - Yes

5. XML documents should be human-legible and reasonably clear.
   - Yes, they are human readable. XML is text, but isn't meant to be read. One advantage of a text format is that it allows people, especially programmers, to look at the data without the program that produced it; in a pinch, you can read a text format with your favorite text editor. Text formats also allow developers to more easily debug applications. Like HTML, XML files are text files that people shouldn't have to read, but may when the need arises. XML isn't meant to be read, it's meant to be machine-processed

6. The XML design should be prepared quickly.
   - The working group was formed in 1996, and the first version of the standard was released in 1998. In the standards world that is considered to be reasonably quick.

7. The design of XML shall be formal and concise.
   - Good question?? Opinions?

8. XML documents shall be easy to create.
   - True if you have an understanding of the problem or issues you want to address, then there is the standard and tools to help you. However, most of us will probably just hire a programmer who knows what they're doing. What this workshop does is allow us to communicate effectively and intelligently with that programmer.

9. Terseness in XML markup is of minimal importance.
   - True, XML can be quite verbose, and it's verbose by design. Since XML is a text format and it uses tags to delimit the data, XML files are nearly always large. That was a conscious decision by the designers of XML. The advantages of a text format are evident, and the disadvantages can usually be compensated at a different level. Disk space is less expensive than it used to be, and compression programs like
zip can compress files very well and very fast.
- We are able to pack in as much meaning as we want. Let's you say as much as you need to say with no limits on space.
- Also done so that anyone can look at code and understand meaning. You can use real words. You don't have to strip words into meaningless acronyms. You can "say" `<scopecontent>` instead of shortening it to `<sc>` which could be misinterpreted or counter-intuitive, or read to mean something else.

Some other things you may want to know about XML:
- XML is for structuring data. XML is a set of rules (you may also think of them as guidelines or conventions) for designing text formats that let you structure your data. XML also works to avoid common pitfalls in language design: it is extensible, platform-independent, and it supports internationalization and localization.
- XML is a family of technologies. XML 1.0 is the specification that defines what XML is. Beyond XML 1.0, "the XML family" is a growing set of tools, standards, and recommendations that offer useful services to accomplish important and frequently demanded tasks. Xlink, XPointer, XSL, XSLT, and XML Schemas are some of those you will hear about today.

In other words, XML is easy to create, easy to read, and designed for use over the Internet.
Comparing SGML, HTML, and XML

Where did XML come from?
[Refer to the table in front of you, it’s easier to read.]

The following table tries to give you an understanding of how these three markup languages:

- Structured Generalized Markup Language (SGML), HyperText Markup Language (HTML), and XML relate to each other.
- Shows you information on the standard, a definition of what the standard does, how the standard is used, and a final description of the markup language.

SGML became a standard in the mid-1980s.
- HTML followed in the 1990s, when web pages and the Internet made such a dent in everyone’s life.
- XML really started to be talked about in the late 1990s.
- You can also see that the size of the standards, in terms of number of pages, has become increasingly less. [For example, 1 ream of paper=500 pages]

By definition

- SGML is a method for creating interchangeable, structured documents. It’s content markup. An international documentation standard for defining descriptions of the structure of different types of electronic documents.
- HTML is a web-based document presentation standard.
- XML is a document processing standard.

So why do we have three standards? Why don’t we hear much about SGML anymore? And why have we moved from talking about HTML to talking about XML?

- SGML has been around since the mid-1980s and some industries have made some significant investments in using it as their standard. There is a significant body of expertise and software to go with it. SGML also became very complex. It’s used for describing thousands of different document types in many fields. It’s used primarily for technical documentation and government applications, and in industries with huge documents, such as medical records, company databases, and aircraft parts catalogs. Because of its size and the way it’s written, **SGML is powerful, it’s just way too complex.**
- Both HTML and XML came from SGML. SGML is complex, so to make it more useful and look better to the users, they developed HTML for presenting data. Unlike SGML, HTML does not hold data itself, it just presents it. HTML is actually an application of SGML, used for formatting and presentation on the web. Most HTML browsers do not support basic SGML constructions, but nearly all SGML authoring tools are capable of producing good HTML documents. **HTML formats for presentation. HTML is too simple: it**
doesn't do enough.

- XML is a public format: it is not a proprietary development of any company. The v1.0 specification was accepted by the W3C as a Recommendation on February 10, 1998. The W3C wanted to create something that was as powerful as SGML, but with the ease of use and acceptance of HTML. XML is not a SGML application, rather it's a set of simple conventions for using SGML without some of the more esoteric features. Known as simplified SGML, XML is a lightweight, cut-down version of SGML which keeps enough of its functionality to make it useful, but removes all the optional features and the more complex and less-used parts of SGML, in return for being easier to write applications for, easier to understand, and more suited to delivery and interoperability over the Web. XML is simplified. It's complex enough to do what we need, and it's simple enough that we can actually do it.

Essentially all three standards offer different levels of difficulty and functionality

- SGML does what HTML and XML do, but it's hard to learn and apply.
- HTML was designed to be functional and easy to use, but it's limited to being used for appearance and presentation.
- In order to do more, increase functionality of SGML, but with the simplicity and ease of HTML, XML was created. Because it has been so recently developed, the developers are still working out the kinks. There are still ways to make it more useful and improvements to the standard that have yet to come.
### Comparing XML and HTML

**Similarities:**
- Both markup languages use tags `<xxx>yyy </xxx>`
- Web browsers display both markup languages
- Both markup languages have W3C working groups

**Differences:**
- HTML tags describe how a document looks
- XML tags describe what a document means
- HTML files can be sloppy
- XML files have to be precise and exact – they have to be well-formed and valid

**In order to better understand XML, compare it to HTML**

**Similarities**
- Both of these standards work under the concept of tags. Tags are made of angle brackets, or greater-than and less-than signs. We see here an opening and a closing tag. Text of the tag itself goes between the angle brackets, and the text the tags are modifying go between the tags.
- Certain versions of web browsers will display both documents, (Internet Explorer version 5.0, and Netscape 7)
- The W3C has been involved in the development of both standards.
- Both HTML and XML are ways of presenting and communicating digital objects. HTML makes digital objects human readable in a browser, and XML is human readable markup of a document that is meant to be machine processed.

**Differences between XML and HTML**
- HTML is used for used for formatting and presentation on the web. That means its tags define what size your header lines are going to be, where the paragraphs are, what your lists look like, whether something is bold, italicized or underlined, etc. The tags tell the browser how to display a document, and some of the things that web page is going to be able to do: display an image, link to another web page, display an interactive form, etc.
  - **Note to instructor:** Write an example such as `<p></p>`, `<b></b>`, `<font = "">`, as opposed to `<title></title>`, `<author></author>`

- XML does not describe how a document will be displayed, although a form of XML can, which we will discuss later today, but that is not the purpose of XML. The purpose of XML is to describe the content of your document (it’s a semantic markup language). The tags in an XML document tell you whether you are looking at a first name or a last name, a title or section or chapter of a document. Not how it should look in a browser.
- Most HTML documents are sloppy, not formatted rigorously. Text is displayed often without closed tags, and in many cases the browser can interpret that a closing tag should be there and displays the document.
  - **Note to instructor:** Write an example such as `<p>`, as opposed to `<title></title>`, `<author></author>`

**XML documents must be both well-formed and valid in order to be displayed and function. They have strict rules to comply with.**

**Well-formed and valid XML**

- **Note to instructor:** customize the examples on this
Well-formed XML: Documents that follow XML tag rules, and each document is a complete, self-contained object.

XML tag rules:

- A root element is required
- Case matters: `<recipe>`, `<Recipe>`, `<RECIPE>` are not the same tag
- No unclosed tags
  Incorrect
  `<title>` Original Nestle Toll House Chocolate Chip Cookies Recipe
  Correct
  `<title>` Original Nestle Toll House Chocolate Chip Cookies Recipe
  `<title>`
- No overlapping tags, they must be properly nested
  Incorrect
  `<Tomato>`Let's call `<Potato>`the whole thing off`</Potato>`
  Correct
  `<Tomato>`Let's call `<Potato>`the whole thing off`</Potato>`
- Attribute values must be enclosed in quotes

Valid XML: When it obeys the rules – the words are in the dictionary and the format is grammatical. In other words, a document that follows both the XML tag rules and the rules defined in its DTD or XML Schema.

What does it mean to have well-formed and valid XML?

- Well-formed XML: Documents that follow XML tag rules, but do not necessarily have a Document Type Definition (DTD) or XML schema (a document which holds the rules for an XML document). Each document is a complete, self-contained object.

XML tag rules: in XML tags are made up. That’s a key part of XML. You make up your own tags, which is very unlike HTML where you have to use standardized and recognized tags to mark-up a document.

- A root element is required. Every XML document must start with a single element that holds the entire document content. In this case the root element of our examples today will be “recipe.” The recipe element may not be repeated again, and all other elements must take place inside the root element. It’s one descriptive word for the entire document - a wrapper around the rest of the tags.
- Case matters: `<recipe>`, `<Recipe>`, `<RECIPE>` all represent different tags, because the use of uppercase and lowercase letters makes a difference
- No unclosed tags. You can get away with all kinds of sloppy stuff in HTML. For example, in most HTML browsers, you can "open" a list item with `<li>` and never "close" it with `</li>`. The browser just figures out where the closing `</li>` should be and automatically inserts it for you. XML doesn't allow this kind of sloppiness. Every open tag must have a corresponding close tag. This is because part of the information in an XML file has to do with how different elements of information relate to one another, and if the structure is ambiguous, so is the information.
- No overlapping tags - A tag that opens inside another tag must close before the containing tag closes. For example, the sequence
  "<Tomato>Let's call `<Potato>`the whole thing off`</Potato>`" isn't well-formed because `<Potato>` opens inside of `<Tomato>` but doesn't close inside of `<Tomato>`.
  The correct sequence must be "<Tomato>Let's call `<Potato>`the whole thing off`</Potato>`".
  In other words, the structure of the document must be **strictly hierarchical**.
- Attribute values must be enclosed in quotes. Unlike HTML, XML doesn't allow "naked" attribute values (i.e., HTML tags like `<TABLE BORDER=1>`, where there are no quotes around the attribute value). Every attribute value must have quotes (`<TABLE BORDER="1">`).

What is valid XML?

- If you have a DTD or schema and you have made your
XML document conform to those rules, then your XML document is considered to be valid. When it obeys the rules – the words are in the dictionary and the format is grammatical. In other words, a document that follows both the XML tag rules and the rules defined in its DTD or XML Schema.
I-15
Exercise: Comparing XML and HTML using a recipe

Part I: What can we do with a recipe?

- **Note to instructor:** The recipe example was chosen because it is non-technical and generic enough for everyone to understand. If you feel it's more appropriate, you may create and present your own example.

First, why are we using a recipe, what is our goal with using this example throughout the day?
- A recipe gives us something familiar to work with as we attack a new topic.
- Recipes are an actual problem that have tangible results. There are a variety of things we can do with a recipe.
- Using a recipe will help us better understand markup in the context of something familiar, it allows us to break something down.

Second, what is a recipe? What information does it give us? (e.g., title, ingredients, process for making something, a photograph, serving information, nutritional information, where to get ingredients or possible optional ingredients or substitutions).

What can one do with the information given to them in a recipe? (e.g., make a grocery list, present the information in a cookbook). XML will allow us to do different things with this information.

Today we'll use the Original Nestle Toll House Chocolate Chip Cookies recipe. **Here it is in Word format.** This is the cookie recipe that you find on the bag of any Nestle Toll House Chocolate Chips.

I-16
Exercise: Comparing XML and HTML using a recipe

Part II: What does a recipe look like in HTML and XML?

- **Note to instructor:** If you choose to replace the earlier recipe example with one of your own, you should customize this example to match.

What can one do with the information given to them in a recipe? (e.g., make a grocery list, present the information in a cookbook). XML will allow us to do different things with this information.

What can one do with the information given to them in a recipe? (e.g., make a grocery list, present the information in a cookbook). XML will allow us to do different things with this information.

Today we'll use the Original Nestle Toll House Chocolate Chip Cookies recipe. **Here it is in Word format.** This is the cookie recipe that you find on the bag of any Nestle Toll House Chocolate Chips.

I-16
Part II: What does a recipe look like in HTML and XML?

- Look at the handout of the recipe marked up in HTML
- Look at the handout of the recipe marked up in XML

Save this one out to use in next Unit.

Notice the differences of the tags we discussed on the last slide.
- What are the different types of markup tags telling us?
  - HTML just describes how something looks. In HTML you need to know the codes to fully understand what the tags are doing to the text.
  - XML tags describe what’s in the tags. In XML the tags are self-explanatory. The XML tags help you
understand the meaning of the text. The XML example is also longer. Remember that terseness doesn’t matter.

• Which is more readable?

Let’s go over the XML tag rules and see where they apply in our handout:

• Declaration: Says using XML version 1.0, which is the only version in existence. Always appears at the beginning of the XML document.
• Root element: <recipe>
• All tags are closed: title, author
• The tags don’t overlap: here we see our background, author example from slide I-14
• Attributes are enclosed in quotation marks: look down at the ingredients section.

Another way to look at XML is by calling things: parent, child, sibling

• In our example background is a parent element to both author and history. Which makes author and history children of background, but also siblings of each other. Find another example? Look at handout of node example to illustrate what you’re talking about

View as displayed in browsers:

• In slide I-16, we see HTML output, how the recipe appears in a browser based on the HTML in the handout I gave you. Very well presented.

I-17
Exercise: Comparing XML and HTML using a recipe
Part II: What does a recipe look like in HTML and XML?
XML as viewed in a browser

I-18
Exercise: A Markup exercise
A joke.

Two North Dakotans come into a bar, slapping each other on the back, laughing, clearly happy as clams. One says to the bartender, "We're celebrating! Give everybody a round on us!"

The bartender says, "So what's the big deal? What are you celebrating?"

And the North Dakotan says, "We just finished a jigsaw puzzle and it only took us four days."

The bartender says, "A jigsaw puzzle? Two

Note to instructor: A joke was chosen for this exercise since it is less intimidating than a technical example. Edit or replace it, if appropriate, to suit your audience’s sensibilities and needs.

{Read joke}

Think about what it is that you want to do with XML, what’s important, what markup do you want to capture?

Split into groups of 3-4 people (or a manageable number of groups) work with those around you and take about 10-15 minutes to mark-up the joke as you see fit.

• What would you mark-up and why?
people? Four days? That doesn't sound like much reason to celebrate."

And the other North Dakotan says, "Are you kidding? The box said '2-3 Years.'"

• Remember that you can make up your own tags, and that there is no one right answer.
• When you're finished we'll discuss what you did, and then share some examples of our own.

[After the participants have presented their markups]

There are different ways to do the markup. What have we seen? By semantic values, by categorizing funny things, such as type of joke, ethnic subjects, punchline

What functions or business purposes are you trying to provide with your markup?
Why did you mark-up the joke the way you did? What do you want to do with the data? What meaning are you assigning to the data?

As you can see, it's a challenge to get different groups to completely agree on what they feel is important to capture. This illustrates how you get to the next step – building consensus, at which point you need to hammer out some consensus and agreement so your new community and partners can all use XML the way you want to.

I-19
Markup examples
<?xml version="1.0"?>
<text>
  <paragraph></sentence
  type="expository">Two North Dakotans come into a bar, slapping each other on the back, laughing, clearly happy as clams.</sentence>
  <sentence type="exclamation">One says to the bartender, <quotation>"We're celebrating! Give everybody a round on us!"</quotation></sentence></paragraph>
  <paragraph></sentence
  type="question">The bartender says, <quotation>"So what's the big deal? What are you celebrating?"</quotation></sentence></paragraph>
  <paragraph></sentence
  type="expository">And the North Dakotan says, <quotation>"We just finished a jigsaw puzzle and it only took us four days."</quotation></sentence></paragraph>
  <paragraph></sentence
  type="other">The bartender says, <quotation>"A jigsaw puzzle? Two people? Four days? That doesn't sound like much reason to celebrate."</quotation></paragraph>
</text>

I-19
Different ways to markup this joke.

☐ Note to instructor: If you chose to alter or replace the joke on the previous slide, edit this slide to match

Straightforward structural markup.
• start by letting you know that it's text.
• identify all the paragraphs and sentences and what types of sentences there are.
• identify quotations.
And the other North Dakotan says, "Are you kidding? The box said '2-3 Years.'"
Why did we do this way? Well people often remember, say, I heard a great joke about a North Dakotan, or, I know a joke and the punchline went like this…

I wanted to keep a record of jokes, these may be some of the things you would search on.

The joke exercise illustrates the different needs, breakdowns, and potentials that arise with XML; it demonstrates what language does, and what you need to make a language work (a vocabulary, a dictionary); and it demonstrates the different functions we will use XML to fulfill.

<table>
<thead>
<tr>
<th>I-22</th>
<th>Key Messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>• A digital object may be created, generated, sent, communicated, received, or stored by electronic means.</td>
<td></td>
</tr>
<tr>
<td>• An electronic record is a specific type of digital object with unique characteristics described by archivists and records managers.</td>
<td></td>
</tr>
<tr>
<td>• In order to preserve content, context, and structure in digital form, we are going to have to move to capturing data, format, and application.</td>
<td></td>
</tr>
<tr>
<td>• One person or organization cannot do it all. You are going to have to make choices.</td>
<td></td>
</tr>
<tr>
<td>• XML is meant to be easy to create, easy to read, and is designed for use over the Internet.</td>
<td></td>
</tr>
<tr>
<td>• eXtensible Markup Language is a means of marking up data, using a specific syntax that you define.</td>
<td></td>
</tr>
<tr>
<td>• XML allows you to share, reuse, and customize data.</td>
<td></td>
</tr>
<tr>
<td>• XML is an international standard supported by the World Wide Web Consortium.</td>
<td></td>
</tr>
<tr>
<td>• Both HTML and XML were derived from SGML.</td>
<td></td>
</tr>
<tr>
<td>• HTML and XML are similar in that they both use markup tags; however, HTML’s markup defines a document’s format, whereas XML’s markup defines a document’s content.</td>
<td></td>
</tr>
<tr>
<td>• Well-formed XML documents follow XML tag rules, and each document is a complete, self-contained object.</td>
<td></td>
</tr>
<tr>
<td>• Valid XML documents obey the rules – the words are in the dictionary and the format is grammatical. In other words, a document that follows both</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>I-22</th>
<th>Digital objects and electronic records are not going away</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Given the number of digital objects in the world, the easy manipulation and deletion of these objects, the obsolescence of storage media on a regular basis, it’s difficult for one person or organization to do it all.</td>
<td></td>
</tr>
<tr>
<td>• Increasingly, we want to be able to share digital objects with ease, and that means being able to have access to other peoples data.</td>
<td></td>
</tr>
<tr>
<td>• Today we’ll focus on one component of a creating a manageable plan: XML as a tool and standard.</td>
<td></td>
</tr>
<tr>
<td>• XML meant to be easy to create, read, and is designed for use over the Internet makes it ideal for use with electronic records.</td>
<td></td>
</tr>
<tr>
<td>• eXtensible Markup Language is a means of marking up data, and that means marking up data in electronic records.</td>
<td></td>
</tr>
</tbody>
</table>

**XML**

- Allows you to share, reuse, and customize data/electronic records.
- Tool that facilitates the interchange, sharing, reuse, customization, automated management, and use of electronic records and information.
- An international standard supported by the W3C.

Both of XML’s predecessors, SGML and HTML have also been used with electronic records.

- XML is like SGML, but much simpler and is designed to be more understandable.
- XML is like HTML in that they both use markup tags in their documents.
- Well-formed XML documents follow XML tag rules, but do not necessarily have an associated/conform to DTD/XML Schema.
- Valid XML documents follow both the XML tag rules and the rules defined in their DTDs/XML Schemas.
the XML tag rules and the rules defined in its DTD or XML Schema.

<table>
<thead>
<tr>
<th>Unit II: What does XML look like?</th>
<th>Unit II: What does XML look like? <em>(Average 48 minutes)</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>II-1</td>
<td>II-1</td>
</tr>
<tr>
<td>This unit includes:</td>
<td>In this unit we'll understand the parts that make up an XML document and what they look like.</td>
</tr>
<tr>
<td>• What is an XML document?</td>
<td>• learn what DTDs and XML Schemas are</td>
</tr>
<tr>
<td>• The metadata and XML relationship.</td>
<td>• technical requirements needed in order to make XML work</td>
</tr>
<tr>
<td>• Document Type Definition (DTD)</td>
<td></td>
</tr>
<tr>
<td>• XMLSchema</td>
<td></td>
</tr>
<tr>
<td>• Namespaces</td>
<td></td>
</tr>
<tr>
<td>• What is involved in using XML?</td>
<td></td>
</tr>
</tbody>
</table>
What is an XML document?


```xml
<?xml version="1.0" standalone="yes"?>
```

Elements: The most basic unit of an XML document. The name of the element (defined by you) should describe the element’s purpose in relation to its contents.

```xml
<recipe>
  <title>Original Nestle Toll House Chocolate Chip Cookies</title>
  <background>
    <author>Ruth Wakefield</author>
  </background>
</recipe>
```

Attributes: Additional data elements that help to more accurately describe an element. Attributes have quotation-mark delimited values that further describe the purpose and content of an element. Information contained in an attribute is generally considered metadata.

```xml
<ingredients>
  <item quantity="1" unit="12 oz pkg.">Nestle Toll House semi-sweet chocolate morsels</item>
</ingredients>
```

In order to build XML applications, you may need to do four things:

- Select or write a DTD or XML schema
- Generate XML documents
- Interprete XML documents
- Display XML documents

All XML documents begin with a **declaration** that declares what version of XML you are using.

- The declaration is always the first thing to appear in your XML document. Currently this is easy since there is only version 1.0, but could become more relevant in the future as the standard evolves.
- Standalone in this reference means that your document does not have an accompanying DTD. Should it have an accompanying DTD, you would reference the location of that DTD here.
- Notice also that the declaration is within angle brackets, this is how all XML notation works, within angle brackets.
- the question mark?=processing instruction <?   ?>. In this case the processing instruction tells the computer we will be using XML version 1.0
- Typically, text within angle brackets, but within another symbol like a question mark or exclamation point means that it’s not your average element. A ? mark signifies a processing instruction. An exclamation mark following by two dashes indicates a comment, <!--       -- >

Elements: Are the most basic unit of an XML document. The name of the element (defined by you) should assign some meaning to the content.

- Elements are what go in the tags that surround your content.
- Notice elements may contain other elements, which are called sub-elements. And, they can also be nested.
- Revisit parent, child, sibling relationships.
- We also see here an example of the **root element**. Remember from the XML tag rules, every XML document must start with a single element that holds the entire document content. In this case the root element is recipe. The recipe element may not be repeated again, and all other elements must be placed between the root element. The root element is one descriptive word for the entire document. Wrapper around rest of tags.

Attributes are additional data elements that help to more accurately describe an element.

- have quotation-mark delimited values that further describe the purpose and content of an element (remember the XML tag rules).
- Information contained in an attribute is generally
considered metadata. Here we see attributes as item quantity and unit.

- Attributes go within the tag, providing further description of the content

You need to have these things in order to have an XML document.

- A declaration,
- elements describing your content,
- possibly attributes modifying your elements.

Go over the XML recipe handout.

☑ Note to instructor: If you chose to substitute another example in place of the recipe, you should consider making the examples in this slide match the XML handout.
The metadata and XML relationship

XML is a meta language. XML is really about adding layers of information to your data, so that the data can be processed, used, and transferred between applications.

Different definitions of metadata:
- data about data
- information about information
- descriptive information which facilitates management of, and access to, other information
- evaluation tool

Metadata assists you with the discovery, description, evaluation, and management of records

The XML and metadata connection
- Elements and attributes are all metadata. They assign meaning to the text within the tags.
- The decision of whether to present your information as attributes or sub-elements will depend on your business needs.

The decision of whether to present your information as attributes or sub-elements will depend on your business needs. What is it you want to do, and what is your ultimate goal in using XML?
- When to use elements and when to use attributes?
  - Sub-elements = nested elements
  - Attributes = additional data elements that occur within an element. May not contain more than one value, but elements can.
  - Both are equally an element
  - Some people feel that they are much less able to manipulate attributes as a unit, and are less easy to expand than elements. And, that they are not as easily tested against a DTD
  - Actions may be taken on sub-elements. If you break the unit down, easier to act upon

Document Type Definition (DTD)

The document which holds the rules that govern what makes an XML document valid. A standard mechanism for defining what elements and attributes may be used in an XML document, where they may appear, and indicating their relationship to one another within the document. In other words, a DTD is the grammar of an XML document.

- A DTD may be internal to an XML document, or external.
  - Internal

On a previous slide we said that if the XML declaration contains a standalone reference, that means it is entirely self-contained. Well, what are the other options? Your XML document may also refer to a Document Type Definition (DTD) or XML Schema.

We've been referring to the need to have some standard for the markup we are using. Why? Because of the need to share information. In order to share information, someone else has to have a way to understand the system you are using, and your systems and computers have to be able to speak to each other – using a common language.

With XML, there have been two methods designed to do this. They function similarly to a dictionary and grammar, because they contain agreed upon rules that
define the meaning of your XML documents and describe how they should be used. The two methods are document type definitions (DTDs) and XML Schema. They serve the same purpose, to define elements and attributes and how they are used, and to define what may be contained in elements. They serve the common purpose of communication.

How do they function? A computer doesn’t understand strings of text, like it would get in a XML document, but with a DTD or XML Schema that defines allowed parameters, a computer knows what to do with the XML document. In most applications, if your XML documents do not conform to a stated DTD or XML Schema you will receive an error and your system will be unable to work with this XML document.

DTD and XML Schema have a similar function, but they have different potentials.

DTD is inherited by XML because it was used with the XML predecessor Standard Generalized Markup Language (SGML), which we talked about earlier. And, DTDs work, but they have limitations.

A DTD
• Declares each part of an XML document and its proper form exactly.
• Defines what tags can go in your document, what tags can contain other tags, the number and sequence of the tags, and the attributes your tags can have.
• The document which holds the rules that govern what makes an XML document valid, so a DTD serves the function of validation.

A well-formed XML document adheres to the XML tag rules. A valid XML document adheres to the tag rules and the rules laid out in a DTD.
• A DTD can prevent an invalid XML documents from getting past the parser (definition to come), so your applications can more or less assume that if they’re handed a document, it meets certain criteria, and will contain what’s expected.

Parser:
• In linguistics, to divide language into small components that can be analyzed. Parsing this sentence would involve dividing it into words and phrases and identifying the type of each component (e.g., verb, adjective, or noun).
• Computer Science: To analyze or separate into more easily processed components.
• A parser is a program that receives input in the form of sequential source program instructions, interactive online commands, markup tags, or some other defined...
interface and breaks them up into parts that can then be managed by other programming. A parser may also check to see that all input has been provided that is necessary.

- A parser will return an error message if an XML document does not comply with the XML tag rules or match to a DTD.

What else should you know about DTDs?
- A DTD may be internal to an XML document, or external. They may be included in the XML document, thereby using the “standalone” reference we saw at the beginning of this unit or they may be external, in which case the “standalone” reference may be replaced by a pointer to the DTD.
- A DTD may be public or private.
  - Private meaning that you or your organization has created it
  - Public DTD is defined by a standards body. This does not refer to access, but ownership.
- DTDs may be confused with Document Type Declarations, because they're spelled similarly, but Document Type Declarations are not called DTDs. A Document Type Declaration is where an XML document states that it wants to be validated against one or more DTDs. The declaration comes after the processing instruction in an XML document, but before the root element.

II-5
Exercise: Document Type Definition of a recipe

```xml
<!DOCTYPE recipe[
  <!ELEMENT recipe (title, background, recipe_info, nutritional_info, comments, ingredients, directions)>
  <!ELEMENT title (#PCDATA)>
  <!ELEMENT background (author, history)>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT history (#PCDATA)>
  <!ELEMENT recipe_info (prep_time, cook_time)>
  <!ELEMENT cook_time (#PCDATA)>
  <!ELEMENT prep_time (#PCDATA)>
  <!ELEMENT nutritional_info (calories, fat, protein, carbohydrates, cholesterol, sodium, fiber)>
  <!ELEMENT protein (#PCDATA)>
  <!ELEMENT calories (#PCDATA)>
  <!ELEMENT carbohydrates (#PCDATA)>
  <!ELEMENT sodium (#PCDATA)>
  <!ELEMENT cholesterol (#PCDATA)>
  <!ELEMENT fat (#PCDATA)>
  <!ELEMENT comments (#PCDATA)>
]
```

II-5
Exercise: Document Type Definition of a recipe to which our XML recipe document complies.

☑️ Note to instructor: If you chose to substitute another example in place of the recipe, you need to make this DTD match.

- Each element and attribute has been declared. They must be declared or the machine won’t recognize the elements or attributes in the XML document when it is processing it. DTDs are necessary for a computer to make use of an XML document. DTDs state what's allowable in XML document, and tells the computer what to expect. DTDs contain the grammar and the rules for our XML.
- PCDATA and CDATA mean the same thing only the former is applied to elements and the latter is applied to attributes. They stand for parsed character data and character data, respectively, and mean that additional element tags are not allowed. In general these elements and attributes will be filled in with text.
- **Required**: means what it says, that attribute is required. Other modifiers that exist are:
  - **default value**, meaning there is a default value that is passed on unless told otherwise
  - **implied**, meaning that the element can omit the
attribute even when no default value is provided
  o fixed, meaning that if that attribute is supplied than it must match whatever the fixed definition is.
  • Also take a look at ingredient or step, we see a little + means 1 or more.
  o A * means 0 or more.
  o A ? means 0 or 1.
  • All of these things constrain or define your data.

What does a DTD do?
  • a grammar (defining the rules of what your XML document can do)
  • a dictionary (defining what can go in your doc), knowledge of the language you will be using in your XML document
  • Helps you to evaluate your XML document and test for validity.

II-6
XML Schema
  Specifies the structure of an XML document and constraints on its content. A schema defines the grammar of an XML document and is for validation.

What are the downsides of using DTDs?
  a. DTDs do not follow XML syntax and semantics
  b. DTDs are mixed into the XML 1.0 specification
  c. No support for Namespaces
  d. DTDs are difficult to extend
  e. No support for schema evolution, extension, or inheritance of declarations
  f. No embedded, structured self-documentation
  g. Defaults cannot be specified separate from the declarations
  h. DTDs cannot specify data types

II-6
XML Schema
  • Approved as a W3C recommendation on 2 May 2001.
  • Provides for more extensive document definition rules than DTDs
  • Specifies the structure of an XML document and constraints on its content.
  • Defines the grammar of an XML document and is for validation.
  • Expresses shared vocabularies and allow machines to carry out rules made by people.
  • Provides a means for defining the structure, content and semantics of XML documents. They do the same thing as DTDs, but are supposed to do it better.

Downsides of using DTDs:
  • DTDs do not follow XML syntax and semantics. Meaning the structure isn't the same: DTDs don't have the same type of tags, and the wording is different
  • DTDs are mixed into the XML 1.0 specification. It would be much less confusing to work with DTDs and XML documents if they were specified separately
  • No support for Namespaces (XML 1.0 was defined before Namespaces) - explained further in the next slide.
  • DTDs are difficult to extend
  • There is no support for schema evolution, extension, or inheritance of declarations: It is difficult to write, maintain, and read large DTDs, and to define families of related schemas
  • There is no embedded, structured self-documentation - comments for example
  • Defaults cannot be specified separate from the declarations (would be convenient to have defaults in separate modules)
  • DTDs cannot specify data types.
What are the benefits of XML Schema?

- XML Schema is expressed in well-formed XML. DTDs are not expressed in XML language.
- XML Schema offers better control over grouping of elements and attributes.
- Allows you to define global element (those that must be used in the same way throughout the XML document) and local elements (those that can have a particular meaning in a particular context).
- Offers an extensive system of datatypes that you can specify for a given element. For example, an element may be an integer, contain a period of time, contain a string, Boolean, a language code, etc. DTDs are unable to restrict character data to a pattern.
- XML Schema supports the use of namespaces.

See the following web site for a list of document type definitions and schemas promoted by every conceivable industry.

http://www.xml.org/xml/registry_search_results.jsp

Benefits of XML Schemas?

- XML Schema is expressed in well-formed XML. DTDs are not expressed in XML language.
- XML Schema offers better control over grouping of elements and attributes.
- Allows you to define a global element (those that must be used in the same way throughout the XML document) and local elements (those that can have a particular meaning in a particular context). Elements as well as attributes can have enumerated values, declarations can be grouped to inherit the same properties and to provide more complex content modeling, and they can also inherit properties.
- Offers an extensive system of datatypes that you can specify for a given element. For example, an element may be an integer, contain a period of time, contain a string, Boolean, a language code, etc. DTDs are unable to restrict character data to a pattern. This makes it easier to describe permissible data content, and to do it more accurately.
- XML Schema also supports the use of namespaces.

Complaints about XML Schema

- Generally too complicated (the specification is several hundred pages in a very technical language), so it is hard to use by non-experts.
- Relatively few applications for schemas available at this point in time.
- The best practices are still evolving.

Why is a schema important for sharing data? For many of the same reasons as a DTD,

- In order to share your XML documents other people have to understand its rules, how it's used, how it functions.
- In order to process it, they need the processing instructions. They need the grammar and vocabulary of your XML.

A DTD is an older (of the two, DTD and schema) system of rules with a peculiar, rather limited syntax.

- Why do we bother to use it? There are two ways to go about creating a grammar, or the rules, for an XML document.
  - DTD, the older version, the original. DTDs were created along with SGML, so they've been around longer and we've inherited them.
  - XML schema, recently created with the benefits that DTDs don't have.
- You have to view it within the context of progress. DTDs were the first way to do things, and they are still valid. People still use them. They needed something to function as the grammar, or rules, and DTDs fulfilled this function until XML schemas were created.
• DTD markup is terse. It can be difficult to read and write, and it is not implemented using XML. DTDs stop well short of providing the kind of control that designers desperately need in the real world.

Benefits of DTDs:
• Many applications already use them
• Schemas are not yet incorporated into many actual applications
• More compact in size, and familiar to those who use them or who have worked with SGML. Also, simplicity. XML schemas can get very large

Why should we use DTDs?
• They’re here and people are working with them.
• Schemas give you all the functionality of XML for sharing, re-using and customizing grammar and dictionary of your mark-up language.
• Allows you to change schemas easily and without affecting the already formatted documents in XML.

You have to decided whether DTDs or Schemas are going to meet your needs. You will also be influenced by which your stakeholders and partners are working with.

II-8
Namespaces

XML namespace: In order for XML documents to be able to use elements and attributes that have the same name but come from different sources, there must be a way to differentiate between the markup elements that come from the different sources.
• XML namespaces are used in XML documents so that elements with the same name, but with different purposes, can be used in the same document.
  For example, one instance of a <TABLE> tag may refer to a data structure with rows and columns while another instance of a <TABLE> tag may refer to a four-legged piece of furniture.
• Appears as “prefix”:“element name” For example, f:table or h:table
• Namespaces can only assure that names are unique and unambiguous. They have nothing to do with document validity.
Namespaces can only assure that names are unique and unambiguous. They have nothing to do with document validity.

The use of namespaces are an important advantage of schemas over DTDs: part of extensibility.

Namespaces prevents the overlap of element or attribute names used by different XML documents/schemas.

### II-9

**Exercise: Schema of a recipe**

**II-9**

**Exercise:** Schema of a recipe. Look at hand out. We see is a schema for a recipe to which our XML recipe document conforms.

☑ **Note to instructor:** If you chose to substitute another example in place of the recipe, you need to make this schema match.

- root element of schema, declares the schema (`<schema>` is the root element of EVERY XML schema)
  - namespace: source that the elements used in the schema are from. Also lets you know that the elements and data types from this ‘namespace’ should be prefixed with xsd:, which you can see throughout the rest of the document
  - `elementFormDefault="qualified"` means that any elements used in the XML document that were declared in the Schema must be qualified in the namespace being used

**Elements and attributes**

- **Elements:**
  - simple – contains data (text, number or date).
    - `string`
  - complex – contains other elements or attributes.
    - `xsd:complexType`
  - `attributes` `xsd:element ref="item"`
  - `name`- unique name of element
  - `content`
    - `empty` (remain empty)
    - `textOnly` (only text, no sub-elements)
    - `eltOnly` (only sub-elements, no text)
    - `mixed` (text or sub-elements)
  - `order`
    - `one` (only one set of elements is allowed)
    - `seq` (elements must appear in the sequence defined by the schema)
    - `many` (elements may appear in any order, any number of times, includes may not appear at all).

- **minOccurs** –
  - 0 (optional, not required)
  - 1 (at least one is required),
    - `maxOccurs`
### What is involved in using XML?

**Equipment**

- **Parser:** Program or class that can read any well-formed XML at its input.
  - Non-validating parser: All XML parsers check the well-formedness of documents.
  - Validating parser: Validating parsers also confirm whether the document is *valid*; that is, that the structure and number of tags make sense.

- **Browser capable of reading/working with XML**
  - Internet Explorer 5.0 (and higher)
  - Mozilla 1.0
  - Opera 5 (and higher)
  - Netscape 7.0

### What is involved in using XML?

What do you need to get started once you have an XML document, and a DTD or schema which define the rules for creating and using that document?

- XML can be complicated because you can make up everything, there’s no standard. You’re making up your own language.
- A DTD or Schema tells your computer what words or strings to use so it knows what to do.
- A parser makes sure your XML document and DTD or Schema match so that your computer can work with the XML document.
- Need a parser that can read XML. There are two types of parsers.
  - Non-validating parsers simply read your XML document while checking for well-formedness. If it’s not well-formed, it won’t read it and will return an error message.
  - Validating parsers check to make sure your documents are valid, that the structure and number of tags make sense.
  - Where do I get a parser? Most tools come with a parser. You will need one at the point of creation, and to subsequently work with a document by validating it.

- You may also need a browser that allows you to read XML, if you’re interested in displaying XML. (Don’t need a browser for databases for example, but you do if you want to display it or put it on the web)
  - IE 5.0
  - Mozilla 1.0 (the people who initially worked on Netscape have moved onto Mozilla)
  - Opera 5 are all browsers with XML reading capabilities.
  - Netscape 7.0 is now also XML compliant.
### Key messages
- An XML document is made up of a declaration, elements, and attributes.
- A DTD holds the rules that govern what makes an XML document valid.
- An XML Schema specifies the structure of an XML document and constraints on its content.
- DTDs and XML schemas have similar functions. DTDs are more widely used, since the XML schema specification is still new.
- In order to use an XML document, you may need to have the following:
  - a parser
  - a browser capable of reading/working with XML

### Go over key messages on slide
- In order to take advantage of all of XML’s benefits (e.g., facilitating the interchange, sharing, reuse, customization, automated management, and use of digital objects and information) you have to have properly formed XML documents.

- Use of XML for electronic records requires an effort/investment, standards allow you to share with others in the long run, facilitates things. Standards mean that you don’t have to reinvent the wheel, help provide structure to use, a method to the madness.

### Unit III: Presenting XML

#### III-1
This unit includes:
- eXtensible Stylesheet Language (XSL)
- XSL Transformations (XSLT)
- XHTML

#### III-2
eXtensible Stylesheet Language (XSL)
- A language for expressing stylesheets.

**Stylesheet:** A definition of a document’s appearance or layout in terms of such elements as default typeface, size, and color of headings and body text, how sections are laid out in terms of space, line spacing, margin widths on all sides, spacing between headings, etc. Typically expressed at the beginning of an electronic document. May be embedded in or linked to a document.

**XSL** is a family of applications:
- XSL Transformations (XSLT)
- XML Path Language (XPath)
- XSL Formatting Objects (XSL-FO)

### Unit III: Presenting XML *(Average 24 minutes)*

#### III-1
- Popular extensions of XML used for presenting
  - These extensions provide more details on the sharing and presenting of XML documents
  - Are from the XML family of standards
  - In order to use XML, you don’t have to use these things, but they provide more options and flexibility

#### III-2
- XML only represents potential, it allows you to assign meaning to text. But, it doesn’t really do anything. You need something else in order to share and use your content. Which is why a family of applications and extensions has been developed for XML. We’re going to look at some of these applications, and we start here with XSL.

**XSL**-(the Extensible Stylesheet Language) is a language that enables stylesheets to be attached to XML documents.
- A language for creating a stylesheet that describes how data sent over the Web using XML is to be presented to the user.

**Stylesheet:**
- A definition of a document’s appearance or layout in terms of such elements as default typeface, size, and color of headings and body text; layout of sections in terms of space, line spacing, margin widths on all sides, spacing between headings, etc. Typically expressed at the beginning of an electronic document. May be embedded in or linked to a document.
- Essentially, a template that describes how something will look.
- Very similar to what HTML does with web documents, and what cascading stylesheets does for HTML. Cascading stylesheets or CSS defines how web pages will look. XSL defines how to create the stylesheets for...
XML.

- XSL is a way to format XML documents for presentation on the web. A way to apply formatting to XML documents without contaminating document content through stylesheets.
- The great potential of XSL is translating XML to HTML (Hypertext Markup Language, the code that builds all of the web pages we see) and for presentation over the Web. A stylesheet describes how various elements in our XML document will appear online.

What happens when you want to change something?

- In XML document, make the change there, then have to adjust stylesheet for how you want it presented.
- If just a style change, make it in the stylesheet and then it is applied to all the documents using that stylesheet with the change being made in one place.
- Contrasts to HTML, documents coded in HTML done separately, one-by-one, stylesheets typically apply to more than one document.

XSL is a family of applications.

- XSL Transformations (XSLT),
- XML Path Language (XPATH)
- XSL Formatting Objects (XSL-FO)

You can apply more than one stylesheet to an XML document.

- For example, both accounting and HR departments want to view the same information differently. Create two stylesheets, one for accounting, one for HR, apply it to an XML document, and it will show up as each department wants it to.

XSL documents use all the processing software of XML, the writing is similar, it’s written in XML syntax.

### III-3

**XSL Transformations (XSLT)**

A language for transforming XML documents. A tool which uses XSL to act on XML documents. XSLT is used to transform XML document contents into something else more suitable for a particular task.

Why would we want to transform a document from one format into another?

- store in one format, display in another
- convert to a more useful format

Example of implementation: Outputting Encoded Archival Description (EAD) documents to HTML for presentation.
on the World Wide Web

- XSLT designed specifically for this task. For example: You can transform your XML data into a Wireless Markup Language (WML – see Unit VI) document suitable for a PDA or cell phone. Or output to the Web in HTML.
- XSLT is the W3C recommendation describing a vocabulary recognized by an XSLT processor to transform information from an organization in the source file into a different organization.

In an XSL transformation

- XSLT processor reads both an XML document and an XSLT style sheet.
- Applies stylesheet to XML document and transforms it into something new.

You can use XSLT to reorder the output according to a specific criteria, to only display certain pieces of information, etc.

A transformation functions similarly to a stylesheet, except that instead of simply applying formatting rules to elements, it can alter the structure of a document to produce a document with a new structure.

- Why transform into another form?
  - store in one format, display in another
  - convert to a more useful format
  - make the document more compact
  - use the document as a front-end to database queries

XSLT is not intended as a completely general-purpose XML transformation language. Rather, it is designed primarily for the kinds of transformations that are needed when XSLT is used as part of XSL.

A transformation expressed in XSLT is called a stylesheet. This is because, in the case when XSLT is transforming into the XSL formatting vocabulary, the transformation functions as a stylesheet.

One good example of implementation of XSLT is outputting Encoded Archival Description (EAD, covered in Unit VII) documents to HTML for presentation on the Web.

### Example: eXtensible Stylesheet Language (XSL) of a recipe

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <head/>
      <body>
        <p>Shopping List for:
        <b><xsl:value-of

### Exercise: XSL of a recipe

- Note to instructor: If you chose to substitute another example in place of the recipe, you need to make this XSL example match.

- What we see is a mixture of XML syntax and HTML coding
- Templates are applied to describe how various parts of the recipe should appear, and they are modified with HTML code.
In this example we want to extract a shopping list from our recipe, and then display it in a browser.

- We can see that it uses some HTML. The document calls in templates which define what happens to different parts of the documents, but instructions for display are described in HTML. The templates say:
  - for example, the title element, which we make bold
  - for example, the item element
  - for example, the attributes quantity and unit
  - each are displayed in a particular order

Really the output is XHTML.

### III-5

**XSLT example**

How does it show up in a browser

- Shopping list of ingredients for our recipe
- Header/Title is a particular size and color
- Ingredients list quantity and item in paragraph form

### III-6

**XHTML**

- XHTML: a reformulation of HTML 4 as an XML application.
  - HTML tags written with XML’s strict syntactic rules

Why should you use XHTML over HTML?

- XHTML is an XML conforming standard. XHTML documents can be used with any general-purpose XML editor, validator, browser, or other program to work on XML docs.
- Documents that follow the stricter XML rules are cleaner, more predictable, and better-behaved in browsers and XML software
- The extensible qualities of XML will benefit XHTML in the long run, making it easier to add new elements and functionality.

Three flavors of XHTML

- Strict: a clean break from current HTML. Many HTML elements deprecated, greater use of CSS
- Transitional: if you want your pages

---

**Note to instructor: If you chose to substitute another example in place of the recipe, you need to make this XML handout match.**

- The pages are also compatible with most HTML browsers in use today.
- XHTML is an intermediary language to help people make the transition from HTML to XML.
- It’s HTML tags written with XML’s strict syntactic rules

Why choose XHTML over HTML?

- With HTML, the browsers do the processing to figure out sloppy code, which takes longer for a page to appear. Strict XHTML makes it easier for machines like PDAs to have browsers that can read HTML files, because these are obviously smaller programs and don’t have to take time or space to figure out the sloppiness.
- XHTML is an XML-conforming standard. XHTML documents can be used with any general-purpose XML editor, validator, browser, or other program to work on XML docs.
to remain compatible with older browsers that do not support stylesheets, retains the elements and attributes of HTML

- Frameset: like strict XHTML with the ability to use frames

- Frameset: like strict XHTML with the ability to use frames

- Frameset: like strict XHTML with the ability to use frames

- Frameset: like strict XHTML with the ability to use frames

There are 3 flavors of XHTML:

- Strict (clean break from current HTML, many elements deprecated [meaning some tags will no longer work], CSS)
- Transitional (want pages to remain compatible with older browsers that don’t support stylesheets, retains the elements and attributes of HTML)
- Frameset (like strict with the ability to use frames)

XHTML is the evolution of HTML

- Web pages become cleaner and will be compliant with more browsers and XML tools.
- Removing style settings will force people to use stylesheets, which will lead to faster development of style support and richer presentation

To reiterate, XHTML is simply HTML following the XML tag rules.

### III-7 Key messages

- XSL, XSLT, and XHTML are extensions of XML used for presentation of documents.
- XSL uses stylesheets to control how XML documents are presented
- XSLT is a language for transforming XML documents from one format into another
- XHTML is a transition from HTML to XML. XHTML is HTML tags written in compliance with XML’s strict syntactic rules

### XML Tools and Editors

**IV-1**

This unit includes:

- A discussion of why you need tools and editors
- A brief look at XML tools and editor that are free

**IV-1**

What do you need to use XML and create XML?

- A parser and a browser capable of reading XML if you want to view data.
- Useful to have software, freeware, or shareware that you can create XML documents with that will help you identify when you have made an error or your

---

State Archives Department, Minnesota Historical Society
01 May 2003
Page 45
document isn’t processing correctly, and to save the document with the correct extension .xml, .xslt, etc.

In this unit we focus on the tools available to you that are free, with pointers to others in Appendix B.

<table>
<thead>
<tr>
<th>IV-2</th>
<th>IV-2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A discussion of why you need tools and editors</strong></td>
<td><strong>You want your work to be easy, especially with XML, otherwise it wouldn’t be worth it. Use tools and editors</strong></td>
</tr>
<tr>
<td>- to create XML tags</td>
<td>- to create XML tags</td>
</tr>
<tr>
<td>- to create different types of XML documents</td>
<td>- to create different types of XML documents</td>
</tr>
<tr>
<td>- to validate XML documents</td>
<td></td>
</tr>
</tbody>
</table>

You can use any text editor to create XML.
- You can use any text editor, such as Notepad, Wordpad, MSWord.
- Except for Notepad, most aren’t free, but they do come with operating systems many of us have on our computers at home or at work.
- You write the document in one of these programs, then change the file extension to reflect the appropriate ending, then use the browser to display.
- What this means is you can use anything to write XML, but you don’t get the XML functionality that the other tools give you.

Often the tools
- Will have drop down lists for you to use,
- Tabs which tell you what document you’re working on, and parsers and validators so that you know your document is well-formed and is working against a DTD or XML Schema.
- Some even create element and attribute lists for you from your DTD or Schema.

What does a tool provide?
- Ability to create varying types of XML documents, XML, DTDs, Schemas, work with XSLT, etc.
- Parsers for validating and checking well-formedness
- Assistance in the creation of tags
### IV-3-4

**Free tools and editors**

**XML Editor:**
- XML Cooktop
  - [http://www.xmlcooktop.com/](http://www.xmlcooktop.com/)

**Online tools**
- Parser
  - Userland Frontier
- Well-formedness and validator

**Other**
- The Apache XML Project
  - [http://xml.apache.org](http://xml.apache.org)
- netbeans.org
  - [http://www.netbeans.org/nonav/index2.html](http://www.netbeans.org/nonav/index2.html)

**URLs with access to many tools:**
- XML.com
- XML Cover Pages
- Information by Lars M. Garshol
  - [http://www.garshol.priv.no/download/](http://www.garshol.priv.no/download/)
- sourceforge.net
  - [http://sourceforge.net/](http://sourceforge.net/)

### IV-3-4

**A brief look at XML tools and editor that are free**

There are not many specific XML tools, such as editors or parsers, that are free.

- The open source community has embraced XML and is adding XML capabilities to the tools they develop. This may mean the tool will include an XML editor or parser, or run on XML, or function with XML, but they are not solely XML editors or parsers. Many of the open-source projects may include XML at some point.

**XML Cooktop**
- XML Cooktop is a development environment for authoring, editing, and testing XSLT style sheets, XML documents, DTDs, and XPATHs. **FREE**

**Online tools**
- you can upload documents to these parsers and online validators

**The Apache XML Project** has a variety of products:
- Xerces - XML parsers in Java, C++ (with Perl and COM bindings)
- Xalan - XSLT stylesheet processors, in Java and C++ ;
- Cocoon - XML-based web publishing in Java
- AxKit - XML-based web publishing, in mod_perl
- FOP - XSL formatting objects, in Java
- Xang - Rapid development of dynamic server pages, in JavaScript
- SOAP - Simple Object Access Protocol
- Batik - A Java based toolkit for Scalable Vector Graphics (SVG)
- Crimson - A Java XML parser derived from the Sun Project X Parser. **All FREE**

**netbeans.org** is a world-class, professional IDE (Integrated Development Environment).
- The NetBeans IDE is the Platform plus modules that include things such as an editor, tools for working with source code (Java, C++ and others), version control, and a lot more. Used to develop code in Java, HTML, XML, JSP, C/C++, and other languages. **FREE**

**URLs with access to many tools, not all are free.** (Go over list on slide)

### IV-5

**Screen shot of XML Cooktop**

**Note to instructor:** *If you and/or your audience is more familiar with another tool, substitute that one for this example.*

- One of the free editors we mentioned, which is used at the Minnesota Historical Society.
If you are familiar with HTML tools, such as Homesite, you'll understand better what editors do. Here we see source, which was checked against the parser that is included with Cooktop, and it returned an error. Tools present you with different options for action. This error that the editor caught on this document is rather important. It demonstrates that many characters we normally use, say in Word, are not legible or acceptable in XML. An editor is useful for helping you to catch these at the point of creation.

**Key Messages**

- XML tools and editors facilitate your work
- There are free tools and editors out there

**Unit V: The XML Family of Standards**

**V-1**

This unit includes:

- XPath
- XPointer
- XLink
- XSL Formatting Objects
- XForms
- SVG
- XQuery

**V-2**

The primary purpose of XPath is to address parts of an XML document.

- XPath models XML as a tree of nodes. XPath helps you get to a point on the tree
- Used to identify particular parts of XML documents.
- XPath functions as part of XSLT, addressing the parts of an XML document that an author wishes to transform.

A more formal definition: XPath is used for addressing parts within an XML document. It’s a declarative language for locating nodes and fragments in XML trees. It’s designed to be used by both XSLT and

**IV-6**

Key Messages

- XML tools and editors facilitate your work
- There are free tools and editors out there

**IV-6**

Go over key messages on the slide

We’ve compiled a list of tools that are not free and it’s in Appendix B.

In this unit we take a look at the expanding family XML-related standards/technologies.

- These technologies are designed to make XML mimic applications we already work with. They provide enhancements to the standard and offer us other capabilities, similarly to what XSL and XSLT did for the presentation of XML.
- We have XML potential, but we need specific tools in order to realize that potential, particularly in terms of tasks we are used to doing in other applications and now want to do in XML.
- Take a look at these at a very brief, high/introductory level.

Note to instructor: Consider customizing this unit to discuss components of the XML family of standards that are relevant to your audience.

*Point to node/tree handout from AM*.
XPointer. It also provides basic facilities for manipulation of strings, numbers and Booleans

V-3
XPointer

- Supports addressing into the internal structures of XML documents.
- Governs the fragment identifier that can go on a URL when you're linking to an XML document, from anywhere
- Specifies a mechanism for pointing to arbitrary chunks (fragments) of a target document.
- Specifies a language that builds upon the XML Path Language (XPath) to support addressing into the internal structures of XML documents.

XPointers enable you to target a given element by number, name, type, or relation, to other elements in the document.

XPointer's extensions to XPath allow it to:

1) be used in URI references to address into resources. Links that point to specific places inside of documents, even when the author of those documents didn't already provide an ID at just the right place.
2) address points and ranges as well as whole nodes. A clear syntax for talking about locations and relationships in hierarchies (such as the structure of XML documents), so that locations are human-readable and writable, rather than mere hash.
3) locate information by string matching. Fine-grained addressing to elements, other information objects, point and strings selections, and spans (also called ranges) inside documents. A way to refer to the many locations that aren't nodes or sets of nodes, such as words, phrases, or other selections, that may even cross element boundaries.

XPointers enable you to target a given element by number, name, type, or relation, to other elements in the document.

A more formal definition: XPointer is for locating internal structures of XML documents. XLinks and URIs can include XPointer parts. XPointer specifies a connection between XPath expressions and URIs. XPointer is based upon XPath. An XPointer expression is basically the same as an XPath expression; although XPath says nothing about URIs. XPointer specifies that connection.

XLink, while useful, only allows you to refer to another document. It is much more common to want to refer to a specific part of another document.

V-4
XLink

- Allows elements to be inserted into XML documents in order to create and describe links between resources.
- Uses XML syntax to create structures that can describe links similar to the simple unidirectional hyperlinks of today's HTML, as well as more sophisticated links.
XLink works by proving global attributes you can use to mark your elements as *linking elements.*

XLink provides a framework for creating both basic unidirectional links and more complex linking structures. It allows XML documents to:

- Assert linking relationships among more than two resources
- Associate metadata with a link
- Express links that reside in a location separate from the linked resources

<table>
<thead>
<tr>
<th>V-5</th>
<th>XSL Formatting Objects (XSL-FO)</th>
</tr>
</thead>
<tbody>
<tr>
<td>V-5</td>
<td>XSL Formatting Objects</td>
</tr>
<tr>
<td></td>
<td>An XML vocabulary for specifying formatting semantics.</td>
</tr>
</tbody>
</table>

- Governs how you insert links into your XML document, where the link might point to anything. Allows for flexible document linking
- Works by providing global attributes you can use to mark your elements as linking elements.
- Provides a framework for creating both basic unidirectional links and more complex linking structures.
- Allows XML documents to:
  - Assert linking relationships among more than two resources
  - Associate metadata with a link
  - Express links that reside in a location separate from the linked resources
- Adds advanced hypertext linking functionality to the Web through:
  - Links that lead to multiple destinations.
  - Bi-directional links (note that "go back" is not at all the same thing: a bidirectional link can be traversed either direction *regardless of whether you went the other way first.*)
  - Links that annotate read-only documents. You can create links that will show up when people view a document even though you don't own the document. This involves a process of deciding whose links you do and don't want to view in this way; but this also makes it possible to build a valuable infrastructure of annotation, commentary, and communal evaluation and discussion on the Web.
  - Links with certain widely-needed special behaviors, such as expand-in-place, new window vs. replacement, automatic follows, and so on. Although these behaviors can be programmed now with Java or other tools, providing the specific semantics means that the behaviors exist as first-class objects themselves, and can be processed in turn (which is not true of general computer programs). Full specifications still require a style language such as XSL, but for many common applications XLink's built-in behavior hints are likely to be enough.
  - Link databases, with all the consequent possibilities for filtering, sorting, analyzing, and processing link collections.

A more formal definition: XLink is a language for typed links between documents. XLink uses XPointer to locate resources.

You use XLink for external documents, and XPath for internal hierarchy.
XSL Formatting Objects: an XML vocabulary for specifying formatting semantics.

- Formatting is the process of turning the result of an XSL Transformation into a tangible form for the reader or listener.
- XSL-FO is a pagination markup language describing a rendering vocabulary capturing the semantics of formatting information for paginated presentation. Essentially, it describes how pages will look when presented to a reader.

XSL-FO is destined to become the preferred stylesheet language for complex formatting.

- Describes how pages will look when presented to a reader.
- Meant to provide sophisticated control over the layout of a document in a desktop-publishing sense, covering things like page margins, foot notes, cross-references, etc.
- Not yet formalized or supported by any browsers
- More specialized to meet publication and design needs.

V-6
XForms

XForms is comprised of separate sections that describe what the form does, and how the form looks. It separates purpose from the presentation.

In the XForms approach, forms are comprised of a section that describes what the form does, called the XForms Model, and another section that describes how the form is to be presented.

XForms uses XML for data transport and HTML for data display. With XForms, the data that are displayed in a form, and the data that are submitted from the form, are transported over the net using XML.

V-6
XForms is comprised of separate sections that describe what the form does, and how the form looks.

- Separates purpose from the presentation.
- In the XForms approach, forms are comprised of a section that describes what the form does, called the XForms Model, and another section that describes how the form is to be presented.
- Uses XML for data transport and HTML for data display. With XForms, the data that are displayed in a form, and the data that are submitted from the form, are transported over the Web using XML.

XForms is an XML application that represents the next generation of Forms for the Web. By splitting traditional XHTML forms into three parts - data model, instance data, and user interface - it separates presentation from content, allows reuse, gives strong typing - reducing the number of round-trips to the server, as well as offering device independence and a reduced need for scripting.

- Purpose: data collection for example
- Presentation: the arrangement of form controls on the screen
- Data: registration information

Business Case for XForms:

- Is not a free-standing document type, but is intended to be integrated into other markup languages, such as XHTML.
- The next generation of Web forms
- The successors to HTML forms
- Richer and more flexible than HTML forms
- Designed to handle interactive transactions
- Designed for integration with XHTML
- Platform-, and device-independent
- Separates user interface from data and logic
- Uses a data model with defined data types and data logic
### V-7

**SVG**
- Uses a user interface with defined data bound controls
- Uses XML and Unicode to exchange data

**Scalable Vector Graphics (SVG):**
- A language for describing two-dimensional vector and mixed vector/raster graphics in XML.
- Allows for three types of graphic objects:
  - Vector graphic shapes (e.g., paths consisting of straight lines and curves)
  - Images
  - Text
- Sends instructions for drawing lines or curves (vectors), and filling these shapes. In that sense describing the image to be created.

**Some benefits of SVG:**
- Can be printed with high quality at any resolution, without the “staircase” effects you see when printing bitmapped images (e.g., GIF or JPEG)
- Text in SVG is selectable and searchable. For example, you can search for specific text strings, like city names on a map.
- Non-proprietary

---

### V-8

**XQuery**

The XQuery language is designed to be a small, easily implementable language in which queries are concise and easily understood. It is also flexible enough to query a broad spectrum of XML information sources, including both databases and documents.

**XML querying is relevant for:**
- Human-readable documents: to retrieve individual documents, to perform context-sensitive searching,
- Data-oriented documents: to query XML representations of databases,
- Mixed-model documents: to perform queries on documents with embedded data

---

**XQuery language (XQuery):**
- Query: a request for information.
- Designed to be a small, easily implementable language in which queries are concise and easily understood.
- Flexible enough to query a broad spectrum of XML information sources, including both databases and documents.

**XML querying is relevant for:**
- Information retrieval.
- Human-readable documents: to retrieve individual documents, to perform context-sensitive searching,
- Data-oriented documents: to query XML representations of databases,
- Mixed-model documents: to perform queries on documents with embedded data
• data-oriented documents: to query XML representations of databases,
  mixed-model documents: to perform queries on documents with embedded data

In short, XML querying is relevant for information retrieval.

• SQL like in appearance and capabilities. Applying what you can do with SQL query to XML. By providing the ability to query XML data regardless of where the data resides.

XML is an extremely versatile markup language, capable of labeling the information content of diverse data sources including structured and semi-structured documents, relational databases, and object repositories. A query language that uses the structure of XML intelligently can express queries across all these kinds of data, whether physically stored in XML or viewed as XML via middleware. Because query languages have traditionally been designed for specific kinds of data, most existing proposals for XML query languages are robust for particular types of data sources but weak for other types. This specification describes a query language called XQuery, which is designed to be broadly applicable across all types of XML data sources.

A query in XQuery is an expression that:
• reads a number of XML documents or fragments
• returns a sequence of well-formed XML fragments

V.9
Key Messages
  • This unit introduced you to some of the standards in the XML family. Other recommendations are constantly emerging.
  • These trends extend XML to perform like other applications.

V-9
Go over key messages on slide.

Why not wait for these new things/something simpler, why do XML now?
• Because some of these things are available to work with now.
• The beauty of XML is that being extensible and flexible, any new improvements should allow you to incorporate these applications in the future.
• New trends constantly emerging. Gives you the high-level needs you may want
• All of these using XML do to more and more precise functions, and they were developed for specific business purposes.

Unit VI: Using XML

This unit includes:
• XML initiatives in a variety of communities.
  o Legal XML
  o MathML
  o Theological ML
  o AgXML
  o Real estate/mortgage XML
  o Wireless ML
  o Minimal ML

Note to instructor: customize this unit to discuss communities and projects of relevance to your audience. Also, consider personalizing this unit with anecdotes, jokes, and stories.

In this unit we’re going to talk about implementation, what it takes to put XML to work.
• Start with non-archival initiatives

XML is popular stuff.
• Lots of different groups with very different needs are exploring the potential of XML.
• Look at each of these with two different goals in mind:
  o 1) we’re going to demonstrate the potential of XML
by examining the different functions and communities it can serve,
  2) we’re going to move into the practical realm by examining how an XML standard is defined and an XML application is implemented.
  • What do you need to bring to the table, who, what do you need to agree on? We’ll take a look at a variety of communities to see the different potentials of XML.

Note to instructor: It is possible to personalize this unit with anecdotes, jokes, and stories.

<table>
<thead>
<tr>
<th>VI-2 Non-archival XML initiatives</th>
<th>VI-2 LegalXML brings legal and technical experts together to create standards for the electronic exchange of legal data.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Legal XML</td>
<td>A member section within Organization for the Advancement of Structured Information Standards (OASIS), the not-for-profit, global consortium that drives the development, convergence and adoption of e-business standards. Members themselves set the LegalXML agenda, using the open OASIS technical process expressly designed to promote industry consensus and unite disparate efforts.</td>
</tr>
<tr>
<td>• Legal and technical experts</td>
<td>Produces standards for electronic court filing, court documents, legal citations, transcripts, criminal justice intelligence systems, and others. OASIS members participating in LegalXML include lawyers, developers, application vendors, government agencies, and members of academia.</td>
</tr>
<tr>
<td>• Organization for the Advancement of Structured Information Standards (OASIS) framework</td>
<td>There are real problems in the legal community. It’s difficult to reach actual consensus with all the different documents: citations, filings, etc.</td>
</tr>
<tr>
<td>• National consensus</td>
<td>XML may become even more important as the nation develops a criminal justice information system for the homeland security effort. There is a real need to share information about terrorists at the different levels (county, state, national) of the criminal justice system around the country.</td>
</tr>
<tr>
<td>• Court filings, court documents, legal citations, transcripts</td>
<td>Private companies, such as WestLaw, are very interested. They could use XML to bring together information from disparate systems to different formats. They use XML for repurposing – print, searchable, online.</td>
</tr>
<tr>
<td>• Criminal Justice Information Systems</td>
<td>Legal XML recently joined with OASIS, they haven’t been together the entire time.</td>
</tr>
<tr>
<td></td>
<td>The legal community has split into two groups and are battling around the role/direction of XML in legal documents.</td>
</tr>
<tr>
<td></td>
<td>The OASIS Legal XML is much less mature developmentally. The splinter group has more robust development and an actual DTD to apply to legal documents.</td>
</tr>
<tr>
<td></td>
<td>A current project from the OASIS group: The Electronic Court Filing Technical Committee will</td>
</tr>
<tr>
<td>VI-3</td>
<td>MathML</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
</tr>
<tr>
<td>W3C working group</td>
<td>MathML is mainly to be used by machines, facilitating the searching and indexing of mathematical and scientific information.</td>
</tr>
<tr>
<td>Expressing math equations in web pages</td>
<td>The goal is to use XML to represent math equations and to allow calculation. Again, extending XML to do things like other applications.</td>
</tr>
<tr>
<td>Graphic representation and calculation</td>
<td>Goal is to enable mathematics to be served, received, and processed on the Web, just as HTML has enabled this functionality for text.</td>
</tr>
</tbody>
</table>

MathML:
- A low-level specification for describing mathematics as a basis for machine to machine communication.
- Provides a much needed foundation for the inclusion of mathematical expressions in web pages.
- An XML application for describing mathematical notation and capturing both its structure and content.

What does MathML do?
- Consists of a number of XML tags which can be used to mark up an equation in terms of its presentation and also its semantics.
- Attempts to capture something of the meaning behind equations rather than concentrating entirely on how...
### VI-4: Theological Markup Language (ThML)

- **Christian Classics Ethereal Library**
- **Subject and Scripture references**
- **Intelligent searching**
- **XSLT: Automatic conversion, variable formats**

**Talking Points:**

The Christian Classics Ethereal Library (CCEL) wants to use XML to enable better citations - references to subjects and scripture - which will better support intelligent searches.

- For automatic conversions - from one format to another. The four gospels, Matthew, Mark, Luke and John, cover much of the same story of the life of Christ, but with significant variations. The synoptic gospels are a format, with each printed side by side, to highlight the similarities and differences in their presentations of the life of Christ. It’s a very common theological practice: medieval works, for example, had parallel columns of original text and commentary.
- XML is going to aid the comparison of similar topics across different texts to allow for a richer and more consistent interpretation of the Christian classics.

The effort has been organized by a rather small community. Not all variants of Christianity are involved - in fact, this effort is completely dominated by Calvin College, so the development of the XML standard is proceeding by fiat.

- The project started as an ecumenical project involving different Christian denominations, and is now a single Protestant college undertaking with a limited application.
- This demonstrates how difficult it is to get different groups to agree, and was beyond the capacity of different Christian denominations.

### VI-5: AgXML

- **Grain and oilseed processing**
- **International markets**
- **Improving business functions**
- **Data models**
- **Promoting adoption of schemas**

**Talking Points:**

Agriculture in the U.S. is really agribusiness.

- In the grain processing industry especially, there are just a few companies around who dominate the international market. Cargill, for example, is based in Minnesota, but has units all over the world. Grain is shipped and processed everywhere, across all sorts of boundaries, including IT systems. Obviously, if the systems spoke the same language, then the data would be much more reliable and far more accessible: that guy in Minneapolis would be able to know exactly what is happening in the Ukraine wheat fields.
- And when you deal in huge volumes on a routine basis, every improvement will generate savings, so AgXML is first of all an effort to determine where an electronic process will improve a business function. There is a very low profit margin. If you increase efficiency, you
increase profits.

- Those functions are examined and their data requirements are modeled.
- Those models generate schemas and then the AGXML consortium promotes their adoption.
- Given the near monopolistic conditions in this area this may well be an example where the XML application creates a community, rather than vice versa. These companies are able to say use this or else, an imposition from above rather than the need to build consensus.

From the web site:

- AgXML is a group of organizations committed to bringing the efficiencies of e-commerce to grain- and oilseed-related business processes. We are accomplishing the following:
  - Identifying business processes that, if electronically-enabled, would improve business-process efficiency and effectiveness.
  - Determining the data requirements of the business processes identified in number.
  - Defining XML schemas and related guidelines that support the data requirements determined in number.
  - Building commitment from participants to integrate XML-based messaging into their business processes and providing a forum for understanding that process.

### VI-6

**Real Estate/Mortgage XML**

- Real estate recording
- Property transactions, mortgages, secondary mortgage market
- MISMO (Mortgage Industry Standards Maintenance Organization)
- PRIJTF (Property Records Industry Joint Task Force)

### VI-7

**Wireless Markup Language (WML)**

- Same content, many devices

**Real Estate/Mortgage XML**

- Multiply the cost of your home by all the property transactions taking place in the country. And multiply the number documents you signed by the same number. That’s a lot of mortgages and a lot of information.
- It’s a very automated industry, but strangely the mode of exchange is still paper format.
- There are a lot of government agencies and private corporations trying to get at that information so they can get at that money. Fannie Mae, for example, does something $400 billion a year - any improvements in the process will generate faster turnover of mortgages and more profits.
- Similar to Legal XML, Real Estate XML has numerous different types of records, and similarly to AgXML there are very low profit margins. If you speed up the process you increase the profits.
- Two national organizations interested in this are MISMO and PRIJTF.

**Wireless ML**

- Addresses that great conundrum of contemporary life: how do we coordinate sharing data across all our different toys? How do you share email across a laptop, PDA, or cell phone? Or a website?
- Different applications
- Broadband and narrowband

There are many different devices and applications involved, as well as a problem with transmission capacities.

You can enjoy a much richer representation of content, context, and structure on a laptop than a pager, for example. But you still want your pager to share information with your laptop. XML and its associated languages (like XSLT) allow for the same content to be shared and appropriately displayed on different machines.

WirelessML is very directed toward reuse and customization.
- The smaller toys have less memory and processing speed, yet everything has to be shared across broadband and narrowband
- WirelessML helps you fit the content for the hardware and software you want to use

WML (Wireless Markup Language)
- A markup language based on XML, and intended for use in specifying content and user interface for narrowband devices, including cellular phones and pagers.
- Designed with the constraints of small narrowband devices in mind. These constraints include:
  - Small display and limited user input facilities;
  - Narrow-band network connection;
  - Limited memory and computational resources.
- Includes four major functional areas:
  - Text presentation and layout - WML includes text and image support, including a variety of formatting and layout commands;
  - Deck/card organizational metaphor - all information in WML is organized into a collection of cards and decks;
  - Inter-card navigation and linking - WML includes support for explicitly managing the navigation between cards and decks;
  - String parameterization and state management - all WML decks can be parameterized, using a state model.

How does it work?
- Functions on the concept of cards. Each card is the equivalent of a screen
- The first card (screen or page) asks the user to make a choice from a menu of 3 items, for example
- The next card is a response page based on the users choice from the previous card
- You navigate through the deck of cards with the use of links

### Minimal XML (MinML)

Potential of XML is that it allows different and disparate groups to develop their own languages, to develop a
- Subset of XML
- Only essential features for data sharing
- Faster parsers
- Simpler information models
- Easier to learn

<table>
<thead>
<tr>
<th>VI-9</th>
<th>VI-9</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key Messages</strong></td>
<td><strong>XML is designed to facilitate reuse and sharing, but it takes a community – process, and tools to make work</strong></td>
</tr>
<tr>
<td>XML has a rich potential</td>
<td>XML has a rich potential</td>
</tr>
<tr>
<td>XML’s potential is realized through the work of specific communities</td>
<td>XML’s potential is realized through the work of specific communities</td>
</tr>
<tr>
<td>It takes a lot of work</td>
<td>It takes a lot of work</td>
</tr>
<tr>
<td>And there are many communities</td>
<td>And there are many communities</td>
</tr>
</tbody>
</table>

Unit VII: Encoded Archival Description

VII-1
This unit includes:
- What is EAD?
- What does EAD do?
- What are the practical components of EAD?
- What makes EAD work?

<table>
<thead>
<tr>
<th>VII-1</th>
<th>VII-2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note to instructor:</strong> You may choose to replace this unit with one that discusses an application of XML of relevance to your audience.</td>
<td><strong>What is EAD?</strong></td>
</tr>
<tr>
<td>EAD is an initiative to work with XML in the archival community that’s been successful.</td>
<td>EAD is a set of rules for designating the intellectual and physical parts of archival finding aids so that the</td>
</tr>
<tr>
<td>We’ll</td>
<td>cover the background of how it started, how it works, and what EAD has accomplished.</td>
</tr>
<tr>
<td></td>
<td>discuss how the EAD project brought people together into a community and what tools are being developed.</td>
</tr>
</tbody>
</table>
Encoded Archival Description (EAD)

The EAD Document Type Definition (DTD) is a standard for encoding archival finding aids. EAD is a set of rules for designating the intellectual and physical parts of archival finding aids so that the information contained therein may be searched, retrieved, displayed, and exchanged in a predictable platform-independent manner.

What’s a finding aid?

{Ask a participant to respond for those who are not archivists}

• Something that points to the material in your repository, describes it, and provides you with the information to decide if you want to use the materials it is describing.
• Anything that assists users in finding your materials.

Before EAD, archivists never had a standard (unlike the bibliographic community which had standards for describing books for decades) - it was very ad hoc.

• Each archive had its own way and approach to the description of records, and the format and look of a finding aid.
• There was no way to take advantage of new technology and share information about holdings, because no one described their collections the same way.

The EAD Project started at the University of California Berkeley Library

• Development of the EAD DTD was a cooperative venture from early on, with specialists at Berkeley working in consultation with experts at other institutions
• Goal was to identify the standard descriptive components of a finding aid
• In 1995 came the initiative to automate finding aids using a standard structure and SGML for exchange – an opportunity to embrace technology. They developed a DTD and in 1998 developed an XML compliant DTD.

Important to note that archival organizations such as the Society of American Archivists and the Library of Congress got behind the project early on to champion the effort and to provide ongoing support.

VII-3

What does EAD do?

EAD provides the:

1. ability to present extensive and interrelated descriptive information found in archival finding aids
2. ability to preserve the hierarchical relationships existing between levels of description
3. ability to represent descriptive information that is inherited by one hierarchical level from another
4. ability to move within a hierarchical

VII-3

EAD provides a standard structure for what has traditionally been an idiosyncratic, unstructured text. What benefits does that actually provide to users?

• EAD provides a standard for descriptive information, and provides and standard structure for traditionally unstructured text.
• Daniel Pitti, the principal investigator for the Berkeley Project, developed requirements for the encoding standard which included the following criteria:
  o ability to present extensive and interrelated descriptive information found in archival finding aids
### VII-4
**What are the practical components of EAD?**

- Standard
- Mechanism for standard refinement and maintenance
- EAD Tag Library
- EAD Cookbook
- EAD workshop
- XSL workshop
- XML editor
- Access mechanism - e.g., printed finding aids, Web delivery, portals

### VII-4
**All the rest of these components make XML and EAD work because consensus is not enough to promote the standard and make it useful.**

- Standard was created
- Mechanism for standard refinement and maintenance was developed – SAA and LoC are essential for this
- EAD Tag Library – a book, like an extended dictionary of the possible tags found in the DTD, was created. It provides information on each element and examples of how it's used
- EAD Cookbook – how to develop a finding aid and put it to use, delivery to the web, and best practices manual
- EAD workshop – education, worldwide and ongoing. You can’t just pick up EAD and use it; you have to learn how.
- XSL – added functionality, how to put the finding aid on the web
- XML editor – a common tool
- Access mechanism - e.g., printed finding aids, Web delivery, portals

EAD is a success story.

- Archivists are usually behind the technology curve, but this time we’re moving right with it.
- This is a full-fledge XML implementation project that has been widely successful and implemented

### VII-5
**What makes EAD work?**

- Business purpose
- Community of interests
- Constant promotion and education

### VII-5
**What makes EAD work?**

- EAD is a tremendous effort. There has been 10 years of work with no end in sight.
- Business purpose –keeping our archival descriptions of our collections viable and available to share
- Community of interests – support and maintenance of interest
- Constant promotion and education – telling people about EAD is ongoing through workshops. Without promotion there’s no sustained interest, and you have to keep the community engaged

### VII-6
**Key Messages**

- EAD is a practical tool for the standardized creation and presentation of finding aids.
- Archivists use EAD because it fulfills a real business need.
- But using EAD is demanding.

### VII-6
**Go over key messages on the slide.**

**Moral:** archivists can use XML successfully.
Creating and supporting the EAD standard was even more demanding.

<table>
<thead>
<tr>
<th>Unit VIII: Putting it all Together</th>
<th>Unit VIII: Putting it all Together <em>(Average 45 minutes)</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>VIII-1</td>
<td>VIII-1</td>
</tr>
<tr>
<td>This unit includes:</td>
<td>XML</td>
</tr>
<tr>
<td>• Learning to speak the same language</td>
<td>• A standard and a tool</td>
</tr>
<tr>
<td>• Business needs; processes and functions</td>
<td>• A technology for data sharing and use</td>
</tr>
<tr>
<td>• Identifying and forming communities</td>
<td>• In order to implement an XML project, you need to start with business needs</td>
</tr>
<tr>
<td>• Developing the application</td>
<td></td>
</tr>
<tr>
<td>• Case study: Minnesota’s Electronic Real Estate Recording Task Force</td>
<td>EAD is a successful project,</td>
</tr>
<tr>
<td>• Implementing your own solution</td>
<td>• helpful to walk through how to make a XML project successful and what process you may have to use.</td>
</tr>
<tr>
<td></td>
<td>• How you make: “The death defying leap from existence in theory to existence in practice”</td>
</tr>
</tbody>
</table>

VIII-2
Learning to speak the same language
A common language needs a:
- Vocabulary
- Dictionary
- Grammar
- And an educational system

A successful XML project needs a:
- Compelling business need
- Collaborative community
- Practical application
- And a very large up-front investment in people, time, money and knowledge

VIII-2
Language is a necessity.
Essentially XML is a second language, which requires
- a vocabulary,
- dictionary,
- grammar,
- education system for people to learn it themselves and teach others how to put it to use.

Similarly, a successful markup language project needs the following components:
- Compelling business need – to be persuasive
- Collaborative community - to work together
- Practical application – to put XML to use
- A very large investment in people, time, money and knowledge – and this isn’t easy, the larger the scale of the project, the larger the investment will need to be
## Business needs

- **Data sharing**
  - Silo metaphor: how does the FBI share data with the CIA and local law enforcement? We know that all those groups are not going to buy the same hardware and software, so we need something that will work independently of any specific architecture. XML is designed to share data across applications and systems.
  
- **Infrastructure independent applications**
  - We cannot change already existing investments, so XML may be the best way to connect across technologies. If technology changes, we want to be able to move from one application to another without data loss and without losing our investment in our information, our knowledge.

- **Web-based transactions**
  - The most logical starting point for sharing data across organizations is use of the web - it’s already there and people already have access to it. XML is designed to work with the web. Because of e-government and e-commerce there is an interest in using the web for automated transactions which will allow more people to do more things online.

- **Improved business processes**
  - There is a lot of information out there, and many communities that need to process more information.

- **Legal mandates**
  - Expectations in meeting our legal mandates may lead us to choose to work with XML.

- **Preservation**
  - And, finally, near and dear to our hearts is preservation. The archives and records management communities have often talked about migration and conversion of records as the way to ensure access and preservation in the face of rapidly changing technology. We must preserve digital information and our investment over time, in spite of obsolescence. XML will certainly foster migration and conversion.

The first concern is having a real application or business need that XML may help fulfill. The second step is developing the appropriate XML language.

## Legal mandates

- **E-Government Act of 2002**

  “4) enterprise architecture
  (A) means
  (i) a strategic information asset base, which defines the mission;
  (ii) the information necessary to perform the mission;

One of the items noted under business needs was “legal mandates.” In government, our business needs are very often defined by legal mandates, in the statutes that define our missions and functions.

Various legislative concerns about developing e-government, the cost of information technology projects, the problems of “doing more with less,” and so on, have resulted in more business needs being translated into specific mandates.

See how XML ties into some specific examples of that by...
(iii) the technologies necessary to perform the mission;
(iv) the transitional processes for implementing new technologies in response to changing mission needs”

“(6) interoperability means the ability of different operating and software systems, applications, and services to communicate and exchange data in an accurate, effective, and consistent manner;”

“(7) integrated service delivery means the provision of Internet-based Federal Government information or services integrated according to function or topic rather than separated according to the boundaries of agency jurisdiction”

Electronic Signatures in Global and National Commerce Act (E-Sign)

“A Federal regulatory agency shall not adopt any regulation, order, or guidance described in paragraph, and a State regulatory agency is preempted by section 101 from adopting any regulation, order, or guidance described in paragraph, unless--

(iii) the methods selected to carry out that purpose do not require, or accord greater legal status or effect to, the implementation or application of a specific technology or technical specification for performing the functions of creating, storing, generating, receiving, communicating, or authenticating electronic records or electronic signatures.”

looking at a federal law, the E-Government Act of 2002, and a federal law, E-sign, that has a counterpart in many states, UETA. Let’s look at some clauses from each law that are worthy of note.

Take a look at this definition of “enterprise architecture.” There are four components and three of them immediately call to mind XML.

- “A strategic information asset base” refers to information being our most important asset.
  - In fact, Minnesota’s technical enterprise architecture starts with the phrase “information is our most important asset.”
- Additionally, we have “the information necessary to perform the mission.”
  - With XML we’re assigning some meaning, some context to our information.
  - XML allows for continued access to our information.
  - It’s additional information tied to performing the mission - let’s just call it executable knowledge.
  - With XML we turn information into knowledge, we add meaning and structure to our knowledge to make it executable knowledge.
- And finally, “transitional processes for implementing new technologies” –
  - what’s the bridge, the middleware to get from one system to another? XML

That’s underscored when we look at interoperability and integrated service delivery. We could have an architecture based on the principle of every agency buying the same hardware and software and using it all the same way, but that’s clearly an impossible dream. So these two principles - interoperability and integrated service delivery - assume that we’ll be working with different operating and software systems, etc., but we’ll still want to share and exchange information and provide services across agencies.

- XML is the best option right now for that.

See it again in E-Sign, the federal equivalent, and UETA, which has been enacted by many states. These laws basically legitimate the use of electronic records. They say that government agencies can specify certain performance measures or functionalities as long as these “do not require … a specific technology or technical specification.”

- This seems to point to XML.
- It’s a standard that doesn’t require any specific hardware or software, because, again, it’s infrastructure-independent.
- It’s knowledge executable in any number of configurations of hardware and software.

What’s the conclusion?

- Business needs and legal mandates are pointing to
Have to remind ourselves that this is a dynamic situation, but XML is right now the best bet to realize the goals these laws describe.

So let's see how this plays out in real life. We'll quickly look at a case study …

<table>
<thead>
<tr>
<th>Collaborative Community</th>
<th>Practical Application</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agreed upon business need</td>
<td>Design</td>
<td>People</td>
</tr>
<tr>
<td></td>
<td>• Data modeling</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Process modeling</td>
<td></td>
</tr>
<tr>
<td>Project sponsor</td>
<td>Business process re-engineering</td>
<td>Time</td>
</tr>
<tr>
<td>Education</td>
<td>Test Assessment</td>
<td>Money</td>
</tr>
<tr>
<td></td>
<td>Delivery Refinement</td>
<td>Knowledge</td>
</tr>
</tbody>
</table>

In order to realize a successful project you need the following things.

- **Collaborative Community** - It's not enough to have a business need, people have to be willing to act upon it.
  - A community interested and committed to working together to address that business need.
  - A project sponsor, because somebody has to take responsibility for all the administrative challenges involved in forming and focusing a community effort.
  - Someone who takes responsibility for education, because there's a whole series of giant learning curves: have to agree there's a business need; have to agree to work together; have to agree that XML is the answer; have to agree on the solution. Education is a continuing challenge, because no community is entirely stable. They include people with nothing better to do, there are attention wanders, etc., so education is a question of bringing people together and keeping them together.
  - The alternative is enforcement - we just make people do this. And we have seen some examples of that with XML. But they are relatively few. It's very unusual to find a situation where one group can just tell a bunch of other groups what to do.

- **Practical Application** - So far, all we've done is talk. We've made some promises, but we have to deliver on that promise, which means delivering a practical application.
  - This really doesn't demand anything different from the routine of application development. You'll have to go through all the usual steps of project management. However, design will be more critically a question of buy or build, because very often you're contemplating something that will talk to your extant system. That is, XML communicates between two already operating systems. You're rarely going to be starting from scratch - you will already have some investment in technology and you'll want to enhance it, build some XML overlay on it so it can do more and talk to other systems.
  - You'll be doing it with partners, so it could get more complex. Any new system is going to demand some business process re-engineering (a business need), so you have to keep this in mind as you go.
into the test phase. The rest seems pretty routine. You move from testing to assessment and then to delivery. Assessment can be extra important because a community may include some very hesitant members - they could want to see something more concrete and tangible before making up their mind to take part. So an assessment that demonstrates the costs and benefits can make the full rollout much more impressive.

- Keep refinement in mind. Nothing is going to remain the same - partners will change, business needs will change, XML will change, technology will change, etc. You’ll need some mechanism in place to allow for the continuing refinement of your applications.

- **Resources** - It all demands resources - lots of them. This is obviously a critical problem. In government, who pays the capital costs? How do agencies with separate budgets share the costs of maintaining a common set of standards? How do you find the time and the people to devote to these efforts in a period of tighter budgets? And how do you hold together a community when the costs and benefits of a system are not equally allocated?

### VIII-6
Case study: MN electronic real estate recording task force

- Task force formed 2000
- Project to end 2004
- Funded by filing fee surcharge
- Private-public partnership
- Entirely voluntary

### VIII-6
Case study: Minnesota Electronic Real Estate Recording Task Force (ERERTF).

Let’s take a look at a case study for some practical lessons in XML development

☑ *Note to instructor – Strongly consider using a local example appropriate for your audience.*

Real estate recording has tremendous potential if it’s automated.

Minnesota is one of the first states to try a statewide standard.

- The task force formed in 2000: A state senator with an interest in IT and the head of the governor’s election campaign (a former Real Estate Lawyer who hated the paperwork) got the Secretary of State to champion the project, and approval from the legislature to form the task force to explore how to automate the system. They did different studies. And, by levying a $1 filing fee surcharge on all closings has made up the $1.2 million funding for the project.
- The project will end in 2004.
- A private-public partnership – everybody who’s interested is invited to join the project (lawyers, archivists, banks, secondary mortgage market, title companies, counties recorders and treasurers)
- Entirely voluntary – can’t force anyone to be at the table
### VIII-7
**Case study: Minnesota Electronic Real Estate Recording Task Force**

**What do we mean by recording and electronic recording?**

**Recording:**
- Recording is the act of entering deeds, mortgages, easements, and other written instruments that affect title to real property into the public record.
- The purpose of recording is to give notice, to anyone who is interested, of the various interests that parties hold in a particular tract of land. Recording determines the legal priority of instruments that affect title to a particular tract of land.

**Electronic recording:**
- A publicly owned and managed county system, defined by statewide standards, that does not require paper or “wet” signatures, and under which real estate documents may be electronically: Created, executed, and authenticated; Delivered to and recorded with, as well as indexed, archived, and retrieved by, county recorders and registrars of title; and
  - Retrieved by anyone from both on- and off-site locations.

### VIII-7
**What do we mean by recording?**

- Recording – as it is now, there is a general standard across the country with minimum variations.
- In Minnesota the County Recorder functions as an archivist; they keep the information permanently.
- Electronic recording will take paper, as the current medium of exchange and electronically connect everyone involved.
- Has to be publicly owned. In Minnesota they can’t get rid of the county recorders because the county system is in the state constitution, but there has to be a standard so that there aren’t 87 (the number of counties in Minnesota) different ways to do it in Minnesota. Need to develop one statewide standard
- And give everyone a piece of the action so the everyone potentially benefits from the project.

### VIII-8
**Case study: Minnesota Electronic Real Estate Recording Task Force**

**What are the business needs?**

- Huge and increasing volume of filings
- Highly inefficient paper workflow between automated activities
- Secondary mortgage market demand for digital records
- Increasing complexity of property rights and descriptions
- Legislative mandate to develop common technical and information architectures

### VIII-8
**What are the business needs?**

- Huge and increasing volume of filings – different property is passed back and forth, and if interest rates fall there are increased filings
- Highly inefficient paper workflow between automated activities – no way to manage more paper more effectively
- Secondary mortgage market demand for digital records - Fannie Mae is a $400 billion/year industry. Time is money. If they can turn over mortgages faster, they can make more money. Time is a big deal, particularly in large counties where they’re unable to deal with the large amounts of paper
- Increasing complexity of property rights and descriptions – transactions take place over property all the time, but you also have to be able to keep track of what rights have applied and do apply over time
- Legislative mandate to develop common technical and information architectures – the state legislature is annoyed that they are continually asked to fund hardware and software implementations year after year without seeing any tangible results. They want state agencies and functions to break down the silos and
develop common technologies that foster information sharing and reuse.

VIII-9
Case study: Minnesota Electronic Real Estate Recording Task Force
What are the options?
- Level 1: images and minimal metadata (scan paper documents and send them in an email message that is manually processed)
- Level 2: images, metadata, digital or digitized signature (scan documents with some substantive metadata that can be automatically processed)
- Level 3: so-called “smart” documents in XML format, following recognized standards (create documents in an XML format which can be entirely processed automatically, populating tract and grantor/grantee indexes, calculating taxes and fees, validating legal descriptions, forwarding approvals and authorizations to different offices etc.)

VIII-9
XML is not the only answer
Minnesota looked at three different options.
- Someone can just scan paper documents and send them in an email message that is manually processed (level 1);
  - Speeds up the transmission process and real benefits are gained primarily in how items are exchanged. It's cheap, and we have not really changed any business processes.
- Scan documents with some substantive metadata that can be automatically processed (level 2);
  - Automate more processes and benefits gained in terms of time
- These two are not technically demanding and can be done without a great deal of investment and re-engineering. Very easy to implement
- Option #3: Create documents in an XML format which can be entirely processed automatically, populating tract and grantor/grantee indexes, calculating taxes and fees, validating legal descriptions, forwarding approvals and authorizations to different offices etc (level 3).
  - “smart documents” are only as smart as the people who design it and the applications that put it to use
  - XML to entirely automate
  - Has the most potential, but requires the most investment of time, people, and money

VIII-10
What are the problems?
Ordinary challenges to re-engineering
- Extraordinary political challenges to re-engineering
- Connectivity to existing systems
- Resources
- Setting standards
- Role of the archivist

VIII-10
What are the problems?
- Ordinary challenges to re-engineering – resistance to change, learning something new, doing something different
- Extraordinary political challenges to re-engineering – the political framework: laws, system, elected officials.
  - Political implications: they can’t cut counties out of middle-management; also elected officials: how many individuals will be driven out of work by efficiency?
- Connectivity to existing systems – XML’s potential is realized by having different applications all work with XML. Currently 87 counties do 87 different processes
- Resources - money, and minimum number of people who understand XML
- Setting standards – agreement is impossible. This is a voluntary process. There are people at the table who don’t work well together and who don’t like each other
- Role of the archivist – a State Archivist is involved because the senator, governor’s campaign manager, and secretary of state were aware that the State Archives was working with technology. They were also aware that the task force is talking about permanent records, even if they don’t come to the State Archives.
<table>
<thead>
<tr>
<th>VIII-11</th>
<th>VIII-11</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>How are we using XML?</strong></td>
<td><strong>How are we using XML?</strong></td>
</tr>
<tr>
<td>Business and workflow analysis</td>
<td>• Didn’t have to use XML, but it offered the most potential</td>
</tr>
<tr>
<td>Data and process models</td>
<td>• Business and workflow analysis - $200000 to do this, huge upfront cost. Studied who creates documents, what do they create, how exchanged</td>
</tr>
<tr>
<td>National standards</td>
<td>• Data and process models – developed from the business and workflow analysis. How the analysis was represented, what was important to capture</td>
</tr>
<tr>
<td>Schemas</td>
<td>• National standards – MISMO and PRIJTF – standards developed outside of any practical application and are not used anywhere. PRIJTF also copyrighted and charges for use. Minnesota standards will be as close to the national standard as possible</td>
</tr>
<tr>
<td>XSLT</td>
<td>• Schemas – not DTDs More effectively able to move between different types of markup</td>
</tr>
<tr>
<td><strong>Case study conclusions:</strong></td>
<td>• XSLT – translate documents from XML into TIFF</td>
</tr>
<tr>
<td>o changed our practices. Talked about XML as a way of becoming independent of technology</td>
<td>• Case study conclusions:</td>
</tr>
<tr>
<td>• However, XML makes Minnesota’s ERER absolutely dependent on collaboration, on a very large scale.</td>
<td>o changed our practices. Talked about XML as a way of becoming independent of technology</td>
</tr>
<tr>
<td>• ERERTF uses XML Spy (mentioned in Appendix B)</td>
<td>• However, XML makes Minnesota’s ERER absolutely dependent on collaboration, on a very large scale.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VIII-12</th>
<th>VIII-12</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>What about your situation?</strong></td>
<td><strong>What about your situation? How do you fill in the components of the plan introduced in Unit I?</strong></td>
</tr>
<tr>
<td>You will have to make choices.</td>
<td>Today we learned about XML, which is a tool, standard, and technology that can help you manage digital objects. XML is a choice, and just one of the many choices you have to make</td>
</tr>
<tr>
<td>The first choice is XML right for me?</td>
<td>We have tried to fill in the resources and education components.</td>
</tr>
<tr>
<td>Variables for the Electronic Real Estate Recording Task Force:</td>
<td>• The workshop is full of URLs and citations of where you can learn more about XML, and we hope you feel better educated about XML as the day draws to a close.</td>
</tr>
<tr>
<td>• Resources: consultants with business analysis and XML expertise, a special surcharge on filing to generate sufficient funds, volunteers with subject matter expertise.</td>
<td>Keep in mind that one size does not fit all. You can’t take one model and set it up –what worked for others may not work for you.</td>
</tr>
<tr>
<td>• Tools: business analysis, communication, negotiation, web sites, facilitated meetings, pilots, subcommittees.</td>
<td>• You need to make decisions about what’s appropriate for you and your partners and environment.</td>
</tr>
</tbody>
</table>
| • Standards/methodologies: XML, XML Schemas, national standard with local extensions. | We have filled in the components here with variables from...
- Education: not enough of it, no common basis of understanding of the choices.
- Technology: web-based transactions, digital signatures, imaging, XML as middleware between legacy applications.
- Partners: banks, realtors, title companies, Fannie Mae, archivists, county officials, state agencies, legislators …

the Minnesota Electronic Real Estate Recording Task Force

{Go over the components.}

☑ Note to instructor: Fill in the components of the equation match your case study.

Let’s take a few minutes now and talk about your situation.
- How do you see yourself using this framework? Or don’t you and why not?
- What are the obstacles? What problems have you encountered? What are the solutions to these obstacles? What are the opportunities?
- What other questions or considerations can you come up with?
- What are your goals?
- What are your resources?
- What do you want to use XML for?
- What is the value of XML to you?
- How do your partners and stakeholders use XML?

Analyze your goals:
- What is your business function?
- Are you meeting that function? Where are the gaps?
- Are you willing to stop doing something you are currently doing in order to do something else?
- Are you willing to reallocate resources, for example for education, if need be?

VIII-13
Key Messages
- In order to speak the same language, we need to have a common vocabulary, dictionary, and grammar. We will also need an educational system to establish, understand, and learn these common factors.
- A successful XML project needs a:
  o Compelling business need
  o Collaborative community
  o Practical application
  o And a very large investment in people, time, money and knowledge
- In Minnesota, the Minnesota Electronic Real Estate Recording Task Force is currently evaluating XML as a solution to automating real estate recording processes.
- In order to implement your own XML project, you will need to consider the following variables:
  o Resources
  o Tools
  o Standards/Methodologies

VIII-13
Go over key messages on the slide.
### Course summary and conclusions

- A review of XML and its applications
  - Digital objects are a fact of life and we need a tool to manage them.
  - Extensible Markup Language is a means of marking up data, using a specific syntax that you define.
  - XML is made up of a declaration, elements, and attributes, and is verified against a DTD or XML Schema.
  - XSLT and XHTML are extensions of XML used for presentation of documents.
  - Free XML tools and editors are available to assist you in your work.
  - There is an evolving family of XML-related standards which extend XML’s functionality.
  - XML’s potential is realized through the work of specific communities.
  - EAD is a practical tool for the standardized creation and presentation of finding aids using XML.
  - You must make choices in order to implement XML within your situation.

### Appendix A
- List of Acronyms

### Appendix B
- XML Tools and Editors that are not free

### Appendix C
- Bibliography