XIA INFORMATION ARCHITECTS CORPORATION

XML
A General Introduction

“If it isn’t on the Web
- it doesn’t exist”

Tim Berners-Lee
Weaving the Web
(1999)

Topics

- What is XML? — Definitions
- What is XML, really? — Fundamentals
- What are the Parts? — The XML Family
- What are the Tools? — Technology
- Where does X ML fit? — Role
- How is XML being used? — Applications
- Where is XML going? — Conclusions

Definitions
What is XML?

XML
Perspectives

Share your Data.
And the tool to carry out Microsoft's version of the future also comes in three words: Extensible Markup Language, or XML.
Steve Ballmer - Microsoft

By year-end 2003, remembering how things were done “before XML” will be as difficult as it is today to remember how they were done “before the web”.
Gartner Group - 2000

XML is rapidly becoming a required standard for electronic business and we consider it a core technology within our Internet platform products
Chuck Rozwat
Executive Vice President
Server Technologies
Oracle Corporation

The two most important things for Microsoft in the new millenium: Windows 2000 and XML.
Bill Gates - Microsoft

XML

Definitions

- ❑ **Extensible**
  - Unlike HTML, XML provides the tools to create new markup vocabularies or extend existing ones
- ❑ **Markup**
  - Descriptive markup scheme based on generic identifiers used to assign names to logical units of content
- ❑ **Language**
  - The grammar to be used in describing document structures where documents are any form of human communication

XML

An XML Document is a self-documented Data Structure

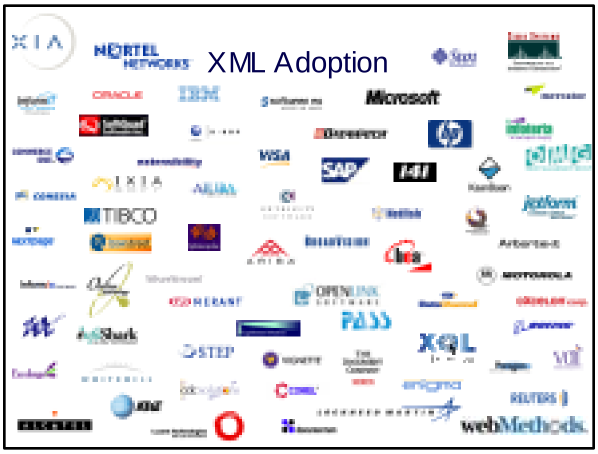
```
<?XML version="1.0" encoding="UTF-8" standalone="yes" ?>
<!DOCTYPE invoice [
<!ELEMENT invoice (to, from, po, amount) >
<!ELEMENT to  (#PCDATA) >
<!ELEMENT from (#PCDATA) >
<!ELEMENT po (#PCDATA) >
<!ELEMENT amount  (#PCDATA) >
<!ATTLIST amount
        currency (USD,CA, EURO) #REQUIRED > ]>

<invoice>
<to>Packet Components</to>
<from>Switches Unlimited</from>
<po>KN00123A-2000</po>
<amount currency="USD">14,599.00</amount>
</invoice>
```

Declared Structural Rules

govern

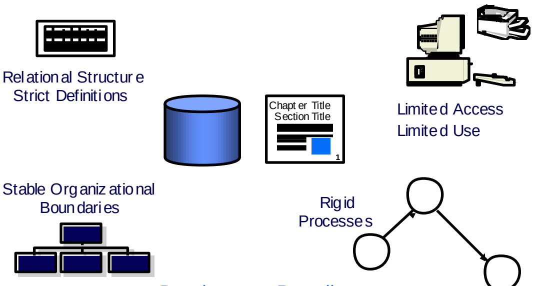
Document Content

XML Adoption



XML Filled a Need

Traditional Systems and Proprietary Formats had failed to meet the challenges of growing complexity

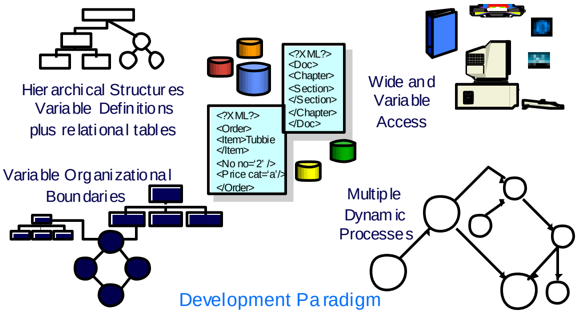


Development Paradigm

Tight Control - Limited Flexibility - Closed Systems

XML: The Open Data Format

XML provided the open and intelligent data format needed by Open Systems



```
<?XML?>
<Doc>
<Chapter>
<Section>
</Section>
</Chapter>
</Doc>
```

```
<?XML?>
<Order>
<Item>Tubbie
</Item>
<No no='2' />
<Price cat='a'/>
</Order>
```

Development Paradigm

Limited Control - Infinite Flexibility - Open Systems

XML

More Specific Definitions

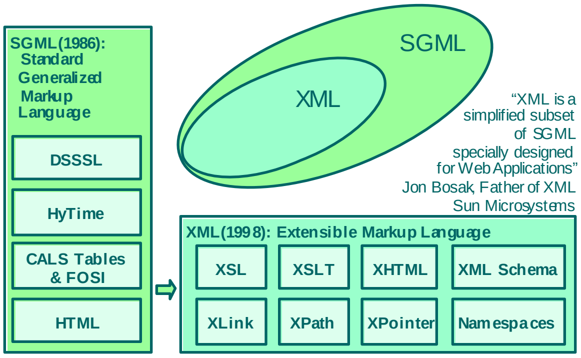
- ❑ The Extensible Markup Language (XML)
  - XML 1.0 is a World Wide Web Consortium
    - Recommendation (February 10, 1998)
    - XML 1.0 Second Edition (October 6, 2000)
  - XML:
    - is a simple and platform-independent method for adding intelligence to interchangeable data
    - is an application profile or restricted form of SGML
  - XML was the result of a long effort to refine and simplify SGML

XML

Definitions

- XML 1.0 states:
  - "XML is a subset of SGML"
  - "XML is an application profile or restricted form of SGML, the Standard Generalized Markup Language"
  - "XML documents are conforming SGML documents."
  - "XML 1.0 specifies a syntax created by subsetting an existing, widely used international text processing standard, SGML, for use on the World Wide Web"
- The Goal of XML
  - "is to enable generic SGML to be served, received, and processed on the Web in the way that is now possible with HTML"

XML and SGML



"XML is a simplified subset of SGML specially designed for Web Applications"
Jon Bosak, Father of XML
Sun Microsystems

So What is SGML?

Definition

- SGML stands for the
  - Standard Generalized Markup Language
- SGML is an international (ISO) standard
  - ISO 8879:1986 *Information Processing - Text and Office Systems - Standard Generalized Markup Language (SGML)*
  - "a language for document representation that formalizes markup and frees it of system and processing dependencies"

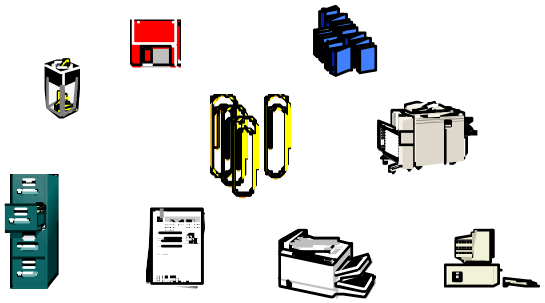
SGML

Introducing Documents to Computers

Publications
Training Manuals
Specifications
Documentation
Reports
Correspondence
Policies
Procedures
Standards
Plans
Directives
Commentaries
Proposals
Business Forms

Document Technologies

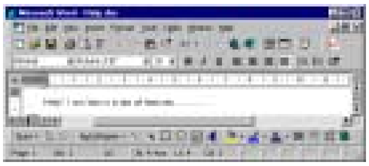
The Truth



The Root of the Problem

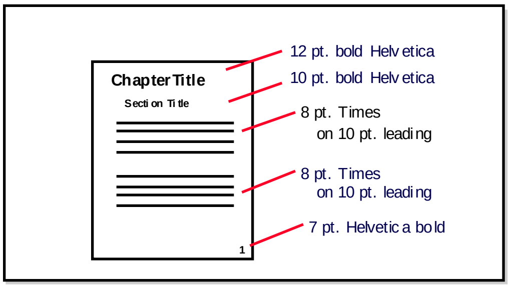
Proprietary Documents

- Traditional electronic documents
  - are produced and maintained in a proprietary and non-intelligent format
  - are simply paper documents in a more reproducible form
  - are printed for use and retention
  - assume a static environment and single format use



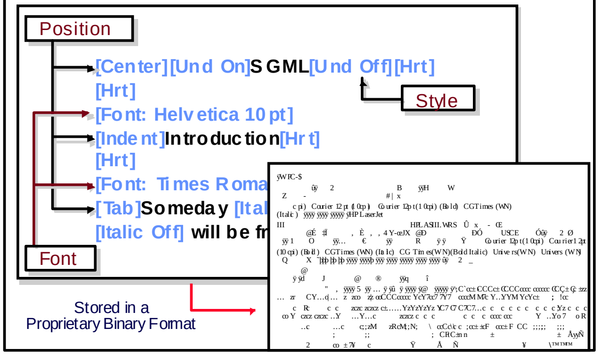
Proprietary Markup

Markup oriented to formatting



Proprietary Markup

Machine Readable Processing Instructions



Position
[Center][Und On]SGML[Und Off][Hrt]
[Hrt]
Style
[Font: Helvetica 10 pt]
[Indent]Introduction[Hrt]
[Hrt]
[Font: Times Roma
[Tab]Someday [Ital
[Italic Off] will be fr
Font
ÿWPC-$
Courier 12pt (10cpi) Courier 12pt(10cpi) (Bold) CGTimes (WN)
(Italic) ÿÿÿÿ ÿÿÿÿ ÿÿÿÿÿ ÿHP LaserJet
HPLASIII.WRS
Courier 12pt(10cpi) Courier12pt
(10cpi) (Bold) CGTimes (WN) (Italic) CG Times(WN)(Bold Italic) Univers(WN) Univers (WN

Stored in a Proprietary Binary Format

Information Evolution

Its not just about Text Documents

❑ Proprietary Encoding

- have restricted the intelligence, interchangeability and value of all information
  - Electronic Data Interchange (EDI)
  - Vector Graphics
  - Product Model Data
  - Metadata, Schema and Relationships
  - Business Forms

❑ Proprietary Encoding

- is a symptom of an older view of the technology world - one that is becoming obsolete

SGML

Created to Free Documents

Document Information

Broadening our definition

❑ A Document

- is a *meaningful* organization of Information
- is *meaningful* because it is communicated between people to achieve specific *goals*
- combines multiple media types together in an organized form that *people* can use
- invokes the generalized structures that underlie the way we communicate
  - this is a fundamental feature of language and cognition

Understanding Documents

How We Read

❑ The Reader of a Document

- scans the layout and format of the contents
- identifies key information items based on the formatting
- determines what kind of document is being read
- determines the rules that apply to this kind of document
- based on these document rules, the meaning of the document can be understood
  - This is a memo giving me instructions (*it impacts me directly*)
  - This is an article from a reputable source (*I can trust its contents*)
  - This is a piece of fiction (*suspend certain expectations*)
  - It is a political announcement (*the opposite is likely true*)

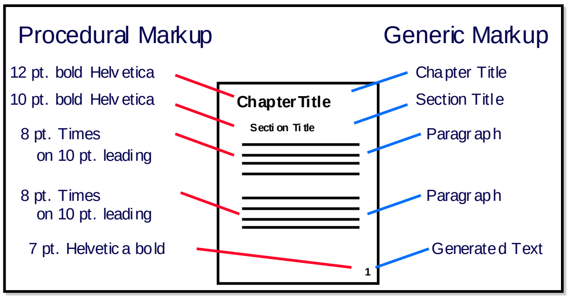
SGML

Fundamental Nature

- ❑ SGML
  - is an attempt to introduce computers to documents and therefore to the way people think and communicate
  - provides the grammar for describing the structure of a particular type, or class, of documents
  - describes a document structure by declaring (or naming) its possible components and their relationships

Generic Markup

SGML is a formalized implementation of Generic Markup



Procedural Markup

12 pt. bold Helvetica

10 pt. bold Helvetica

8 pt. Times on 10 pt. leading

8 pt. Times on 10 pt. leading

7 pt. Helvetica bold

Generic Markup

Chapter Title

Section Title

Paragraph

Paragraph

Generated Text

The SGML Experience

SGML allowed complex documents to be processed

```
<!DOCTYPE SB PUBLIC "-//ATA-BOEING//DTD SB-BOEING-VER2-LEVEL3//EN" []>
<SB MODEL="757"DOCNBR="757-75-0005" SPL="81205" TSN="0" OIDATE="19940106"
REVDATE="0" CHAPSECT="7532" CHAPNBR="75" SEQNBR="0005"
SBTYPE="STANDARD" CHG="N" LANG="EN" REGACT="NO">
<TITLE>
AIR - COMPRESSOR BLEED CONTROL - ENGINE BLEED VALVE EICAS STATUS
MESSAGE SIGNAL REMOVAL - RB211-535 ENGINES
</TITLE><SBFMATR CHG="N" KEY="SBFMATR218">
<TITLE>
Summary
</TITLE><SBFMSECT CHG="N" KEY="SBFMSECT231">
<TITLE>
BACKGROUND
</TITLE>
<PARA>
This service bulletin gives instructions to remove the engine stability bleed valve
control unit signal which causes the (L), (R) ENG SURG CONT status message on the
engine indication and crew alerting system (EICAS).
</PARA>
```

SGML

Introduced the Idea of a Document Type Definition

- ❑ SGML:
  - needed a means for formalizing markup so that computers could:
    - check the validity of a file
    - automatically manipulate the contents of a document based on rules built into an application
  - introduced the idea that there would be a computer-readable set of rules that would apply to a type of documents
- ❑ ISO 8879 defines a Document Type Definition as:

  “Rules, *determined by an application*, that apply SGML to the markup of documents of a particular class”

The SGML Experience

The Outcome

- ❑ The SGML Standard
  - Allows tag minimization
    - reducing keyboarding costs and file size
  - Allows fundamental features to be adapted
    - permitting customization to meet application requirements
  - Provides a highly flexible language for declaring document components and their relationships
    - Exceptions in the DTD Rules
    - An array of Declared Values for Attributes
    - the AND connector
- ❑ These components were difficult to implement
  - Developing SGML applications was very expensive

The SGML Experience

The Outcome

- ❑ SGML
  - Was implemented widely within the Aerospace and Military sectors
    - where money was not the main consideration
  - Was not leveraged to address problems with other document technologies
    - EDI
    - Graphics
    - Product Model Data
    - Metadata, Schema and Relationships
    - Business Forms

World Wide Web

Project Charter 1989

- ❑ Objective
  - “to allow information sharing within internationally dispersed teams”
- ❑ Requirements
  - “Integrate Information from a variety of systems”
  - “Provide a simple, common interchange format”
  - “Permit inexpensive viewers”
  - “Allow information to be accessed by all hardware and software platforms”
  - “Permit keyword searching”
  - “Emphasize link navigation for finding information”

World Wide Web

HyperText Markup Language (HTML)

- ❑ SGML
  - used to create HTML Document Type Definition (DTD)
  - “SGML is a standard in Hypertext circles” T. Berners-Lee
- ❑ HTML proved
  - a simple SGML application could support a universal requirement to share information
  - that the full complexity of SGML was not necessary
  - that HTML could not adapt to meet all requirements with only one set of tags
  - the use of the HTML DTD was relaxed and prone to error

The HTML Experience

The Imperfect Solution that Changed the World

```
<HTML>
<HEAD><title>XML</title><HEAD>
<body bgcolor="#FFFFFF" leftmargin="0" topmargin="0" >
<TABLE BORDER="0"CELLPADDING="0" CELLSPACING="0" WIDTH="590" >
 <TR><TD COLSPAN="4" VALIGN="TOP">
   <P><FONT FACE="Verdana, Arial, Helvetica, sans-serif" SIZE="-1">XML stands
     for the Extensible Markup Language. Formally XML is a World Wide Web Consortium
     (W3C) Recommendation dated February 10, 1998. </FONT></P>
   <P><FONT FACE="Verdana, Arial, Helvetica, sans-serif" SIZE="-1">XML is the result of
     an effort on the part of the W3C to specify "a simplified subset of SGML specially
     designed for Web applications. This subset, called XML (Extensible Markup
     Language), retains the key SGML advantages of extensibility, structure,
     and validation in a language that is designed to be vastly easier to learn,
     use, and implement than full SGML.. ..XML has been designed for maximum
     expressive power, maximum teachability, and maximum ease of implementation."
     <I><A HREF="http://sunsite.unc.edu/pub/sun-info/standards/xml/why/xmlapps.htm">
     XML, Java and the Future of the Web</A></I>, Jon Bosak, 1997. </FONT></P></TD>
 </TR></TABLE></body></html>
```

XML

The Answer

- ❑ XML
  - Offers the ability to create new tag vocabularies
    - Hence “Extensible” Markup Language
  - Constrains the features of SGML
    - simplifies processing (everything that created complex ambiguity is removed)
    - opens the door to broad application support (something SGML never enjoyed!)
    - makes universal browser support possible (allowing browsers to only look at the instance)
    - makes it attractive to programmers (passing the desperate Perl hacker test)
    - maintains the assurance that correctness can be enforced (no more error-handling code)

XML

Fundamentals

SGML and Correctness

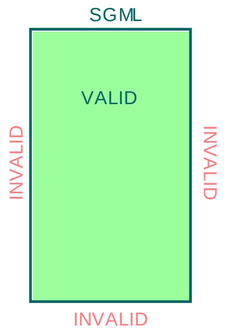
An Unforgiving Model

VALID

An SGML Document must always have a valid DTD and must always be valid under the rules of that DTD.

INVALID

An SGML Document that is not valid according to the applicable DTD receives no further processing (a fatal error)



FOCUS:
Ensuring
Validity
and
Portability

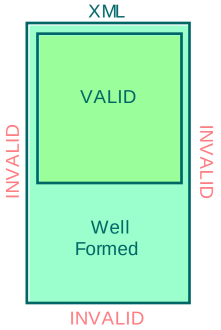
XML and Correctness

A More Flexible Model

Well Formed
All XML documents must be well-formed meaning that the markup is clear and easily processed.

VALID
The XML Document is well-formed and valid under the rules of the applicable DTD.

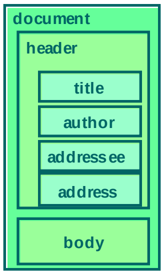
INVALID
The XML Document is not well-formed and this constitutes a fatal error.



FOCUS:
Enabling
Efficient
Processing

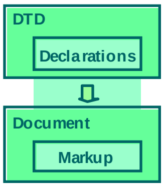
Well-Formed XML Documents

- ❑ A Well-Formed Document:
  - implements the base requirements as set out in the XML Recommendation
  - follows these general parameters:
    - There is a single root element that contains the complete contents of the document
    - All Elements must have start and end tags
      - no omission of end tags
      - empty elements are the only exception
    - All Elements are neatly nested
      - no overlapping of elements
    - All Entities referenced within a document must be declared in the DTD
    - All Attributes must be in quotation marks



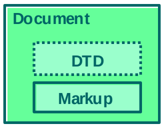
Valid XML Documents

- ❑ A Valid Document:
  - is a Well-Formed document
  - has a Document Type Definition (DTD)
    - defines every element, attribute and entity used in the document
  - follows the rules set out in the DTD
  - has a root element with a name that matches the name in the Document Type Declaration
    <!DOCTYPE invoice [*declarations*]>
    <invoice>*content*</invoice>



Standalone Documents

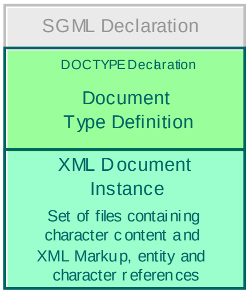
- ❑ A Standalone Document:
  - is a document that can be processed without the need to access an externally stored DTD
  - can therefore be a Well-Formed document for which there does not exist an applicable DTD
  - can also be a valid document where the DTD is included within the document
    - The DTD is stored in the internal subset



The XML Document

Made up of Two Parts: the Rules and the Content

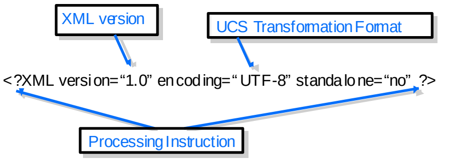
- SGML Declaration
  - defines Syntax, Features, character sets, & processing parameters
  - Not used in XML
- A Document Type Definition
  - defines the rules that will govern a class of documents
  - linked to an Instance through the DOCTYPE Declaration
- A Document Instance
  - Data and Markup


SGML Declaration
DOCTYPE Declaration
Document Type Definition
XML Document Instance
Set of files containing character content and XML Markup, entity and character references

XML Declaration

Just could not live without it

- ❑ Optional XML Declaration
  - "XML documents may, and should, begin with an XML declaration which specifies the version of XML being used."

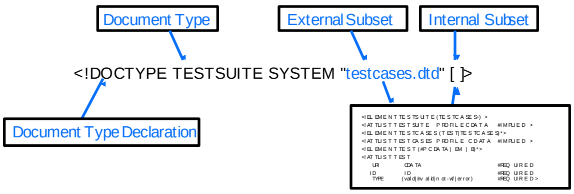


<?XML version="1.0" encoding="UTF-8" standalone="no" ?>

UCS - Universal Character Set

Document Type Declaration
Definitions

- XML 1.0
  - "The XML document type declaration contains or points to markup declarations that provide a grammar for a class of documents. This grammar is known as a document type definition, or DTD."



Document Type

External Subset

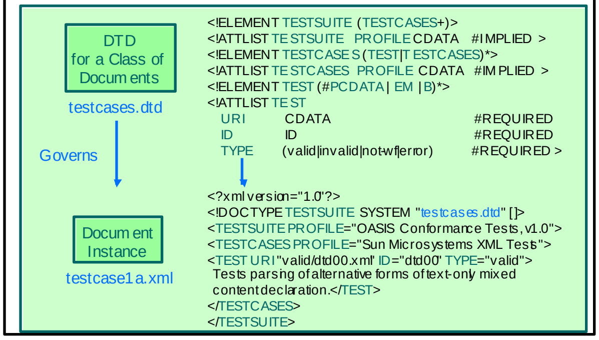
Internal Subset

```
<!DOCTYPE TESTSUITE SYSTEM "testcases.dtd" [ ]>
```

Document Type Declaration

```
<!ELEMENT TESTSUITE (TESTCASES+) >
<!ATTLIST TESTSUITE  PROFILE CDATA   #IMPLIED >
<!ELEMENT TESTCASES (TEST|TESTCASES)*>
<!ATTLIST TESTCASES  PROFILE  CDATA  #IMPLIED >
<!ELEMENT TEST (#PCDATA | EM | B)*>
<!ATTLIST TEST
  URI       CDATA                          #REQUIRED
  ID        ID                             #REQUIRED
  TYPE      (valid|invalid|not-wf|error)   #REQUIRED>
```

Document Type Definitions
Rules that govern a class of documents



DTD for a Class of Documents

testcases.dtd

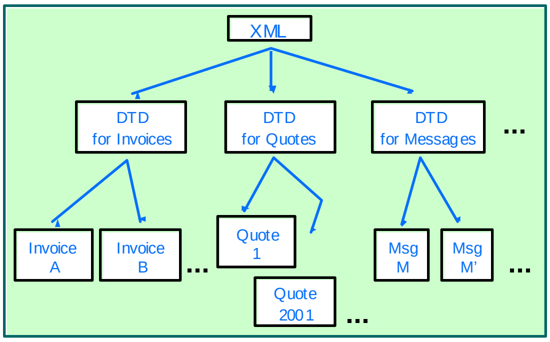
Governs

Document Instance

testcase1a.xml

```
<!ELEMENT TESTSUITE (TESTCASES+)>
<!ATTLIST TESTSUITE  PROFILE CDATA   #IMPLIED >
<!ELEMENT TESTCASES (TEST|TESTCASES)*>
<!ATTLIST TESTCASES  PROFILE CDATA  #IMPLIED >
<!ELEMENT TEST (#PCDATA | EM | B)*>
<!ATTLIST TEST
  URI       CDATA                          #REQUIRED
  ID        ID                             #REQUIRED
  TYPE      (valid|invalid|not-wf|error)   #REQUIRED >

<?xml version="1.0"?>
<!DOCTYPE TESTSUITE SYSTEM "testcases.dtd" []>
<TESTSUITE PROFILE="OASIS Conformance Tests, v1.0">
<TESTCASES PROFILE="Sun Microsystems XML Tests">
<TEST URI "valid/dtd00.xml" ID="dtd00" TYPE="valid">
 Tests parsing of alternative forms of text-only mixed
 content declaration.</TEST>
</TESTCASES>
</TESTSUITE>
```

XML, DTDs and Instances
Having different Markup Languages is the Point



XML

DTD for Invoices

DTD for Quotes

DTD for Messages

...

Invoice A

Invoice B

...

Quote 1

Quote 2001

...

Msg M

Msg M'

...

Document Type Definition
Declaring the Rules

- The DTD

  Rules, *determined by an application*, that apply XML to the markup of documents of a particular class

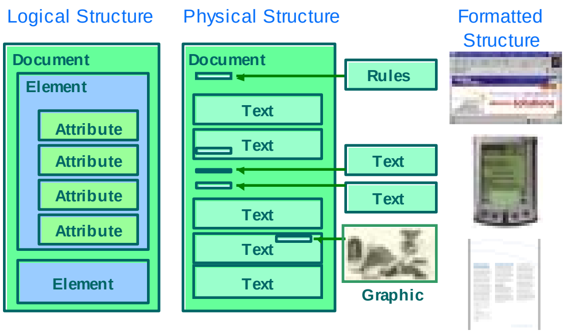
Logical

- Elements - logical units of data
- Attributes - properties associated with elements

Physical

- Entities - physical units of data
- Notations - the format or data type of units of data

XML and Structure
Separating the Different Views of Information



Logical Structure

Physical Structure

Formatted Structure

Document

Element

Attribute

Attribute

Attribute

Attribute

Element

Document

Rules

Text

Text

Text

Text

Text

Text

Text

Text

Graphic

Logical

Element Declarations
Building a DTD

- Each element within a DTD must be declared using the following syntax:

  <!ELEMENT identifier content >

- The content of an Element can be:
  - a "content model" that describes the content of the element

    or
  - explicitly "declared" as a particular kind of content

Logical

Element Identifiers

Naming Elements

- Element Name
  - Begins with a letter, an underscore (_), or a colon (:), and may additionally contain digits, periods (.) and hyphens (-)
  - Examples of Element Names

title	(Element type - *Title*)
body	(Element type - *Body*)
para	(Element type - *Paragraph*)

```
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Warning (#PCDATA)>
```

- Element Names are defined by the users of the data

Logical

Element Declarations

Content Model

- The Content Model of an Element Declaration can describe:
  - Character data that might contain XML Markup
  - Other elements (structure)
  - A mixture of data and elements
- The Content Model consists of:
  - A primary Model Group that is required

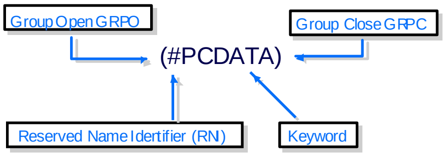
```
<!ELEMENT identifier (model group) >
```

Logical

Model Group for Data

#PCDATA

- The Model Group for Data that may contain XML Markup (i.e. entities and elements) is:



Group Open GRPO

Group Close GRPC

(#PCDATA)

Reserved Name Identifier (RNI)

Keyword

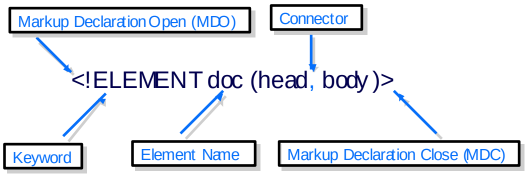
Examples: <!ELEMENT date (#PCDATA)>
<!ELEMENT note (#PCDATA)>

Logical

Model Group for Structure

Elements within Elements

- The Model Group for Structure is:



Markup Declaration Open (MDO)

Connector

<!ELEMENT doc (head, body)>

Keyword

Element Name

Markup Declaration Close (MDC)

The Element “doc” is made up of an Element “head” followed by an Element “body”

Logical

Model Group for Structure

Connectors

- The order of elements within a Model Group is determined by Connectors:
  - **,** **Sequence** connector
    Elements must occur and in the order indicated
  - **|** **OR** connector
    One and only one of the elements can occur (a choice)

Logical

Model Group for Structure

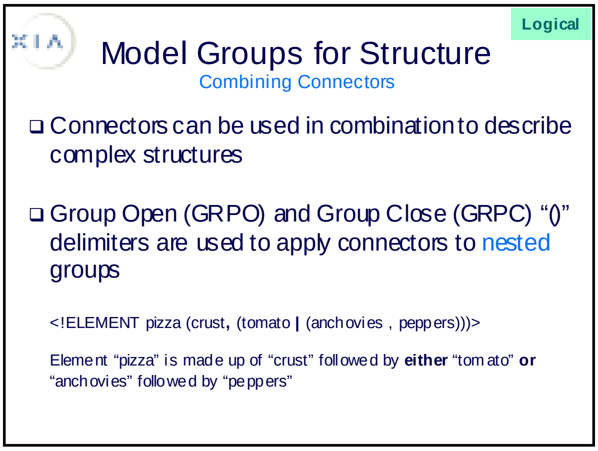
Using Connectors

- Sequence Connector

```
<!ELEMENT play (beginning, middle, end) >
```

- OR Connector

```
<!ELEMENT turn (left | right) >
```



Logical

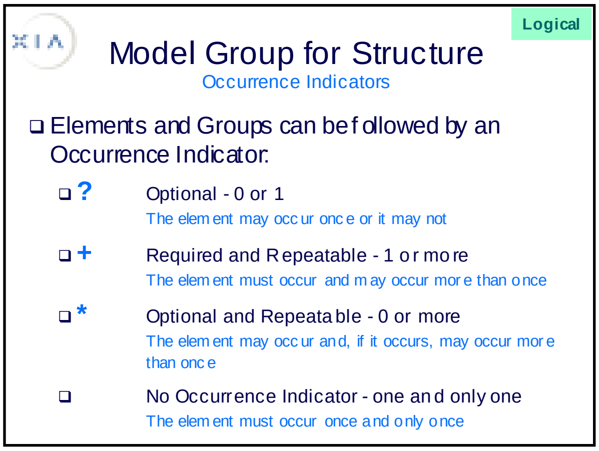
Model Groups for Structure

Combining Connectors

- ❑ Connectors can be used in combination to describe complex structures
- ❑ Group Open (GRPO) and Group Close (GRPC) "()" delimiters are used to apply connectors to nested groups

<!ELEMENT pizza (crust, (tomato | (anchovies , peppers)))>

Element "pizza" is made up of "crust" followed by **either** "tomato" **or** "anchovies" followed by "peppers"

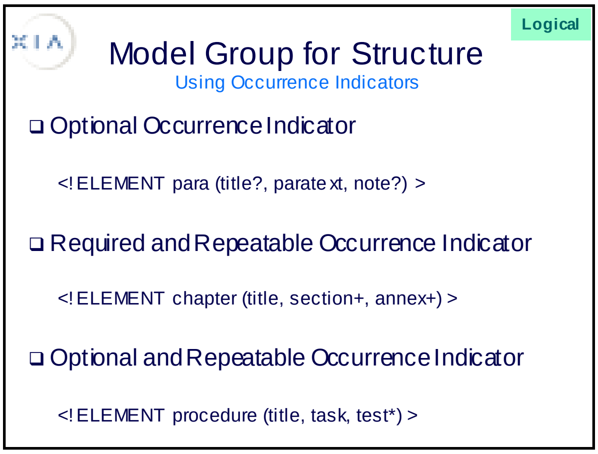


Logical

Model Group for Structure

Occurrence Indicators

- ❑ Elements and Groups can be followed by an Occurrence Indicator:
  - ❑ **?** Optional - 0 or 1
    The element may occur once or it may not
  - ❑ **+** Required and Repeatable - 1 or more
    The element must occur and may occur more than once
  - ❑ ***** Optional and Repeatable - 0 or more
    The element may occur and, if it occurs, may occur more than once
  - ❑ No Occurrence Indicator - one and only one
    The element must occur once and only once



Logical

Model Group for Structure

Using Occurrence Indicators

- ❑ Optional Occurrence Indicator

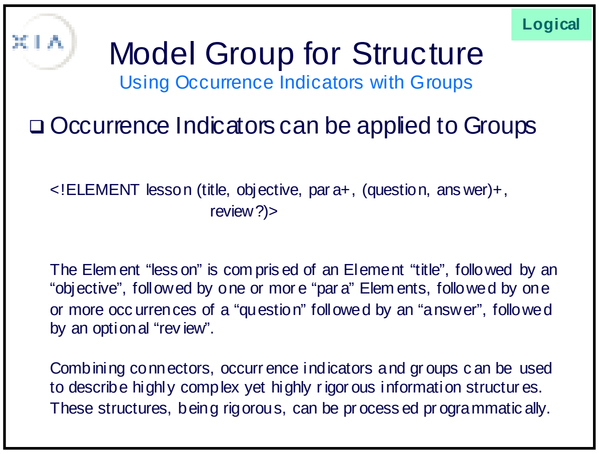
  <!ELEMENT para (title?, paratext, note?) >

- ❑ Required and Repeatable Occurrence Indicator

  <!ELEMENT chapter (title, section+, annex+) >

- ❑ Optional and Repeatable Occurrence Indicator

  <!ELEMENT procedure (title, task, test*) >



Logical

Model Group for Structure

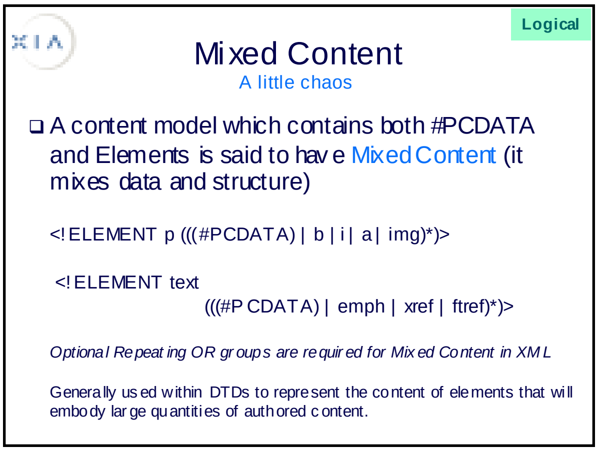
Using Occurrence Indicators with Groups

- ❑ Occurrence Indicators can be applied to Groups

<!ELEMENT lesson (title, objective, para+, (question, answer)+,
review?)>

The Element "lesson" is comprised of an Element "title", followed by an "objective", followed by one or more "para" Elements, followed by one or more occurrences of a "question" followed by an "answer", followed by an optional "review".

Combining connectors, occurrence indicators and groups can be used to describe highly complex yet highly rigorous information structures. These structures, being rigorous, can be processed programmatically.



Logical

Mixed Content

A little chaos

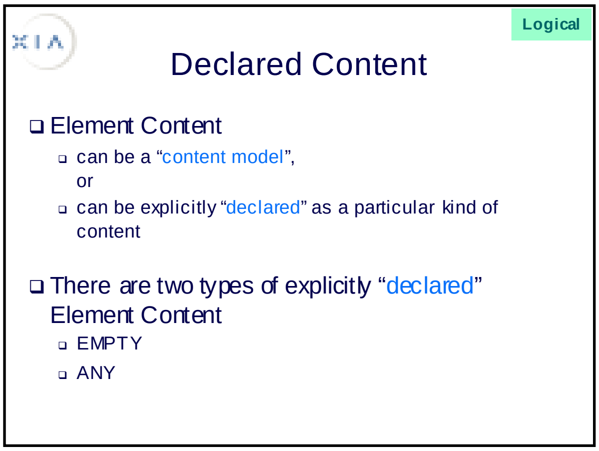
- ❑ A content model which contains both #PCDATA and Elements is said to have Mixed Content (it mixes data and structure)

<!ELEMENT p (((#PCDATA) | b | i | a | img)*)>

<!ELEMENT text
(((#PCDATA) | emph | xref | ftref)*)>

Optional Repeating OR groups are required for Mixed Content in XML

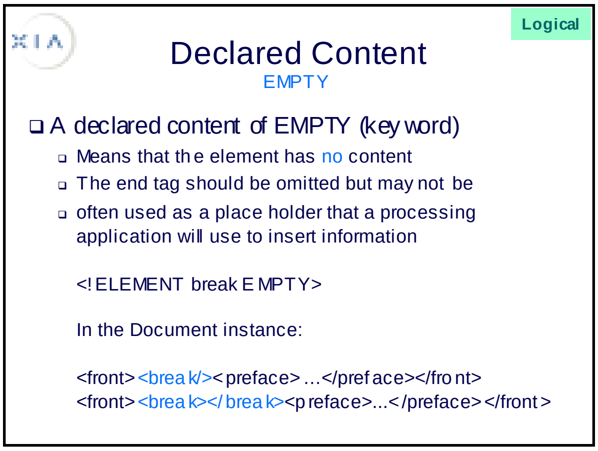
Generally used within DTDs to represent the content of elements that will embody large quantities of authored content.



Logical

Declared Content

- ❑ Element Content
  - ❑ can be a "content model",
    or
  - ❑ can be explicitly "declared" as a particular kind of content
- ❑ There are two types of explicitly "declared" Element Content
  - ❑ EMPTY
  - ❑ ANY



Logical

Declared Content

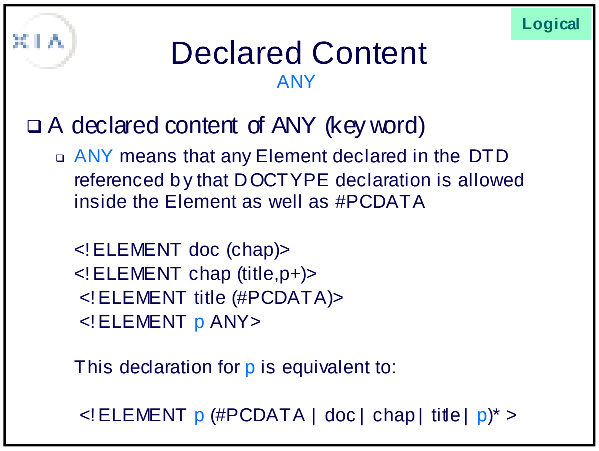
EMPTY

- A declared content of EMPTY (keyword)
  - Means that the element has no content
  - The end tag should be omitted but may not be
  - often used as a place holder that a processing application will use to insert information

    <!ELEMENT break EMPTY>

    In the Document instance:

    <front><break/><preface>...</preface></front>
    <front><break></break><preface>...</preface></front>



Logical

Declared Content

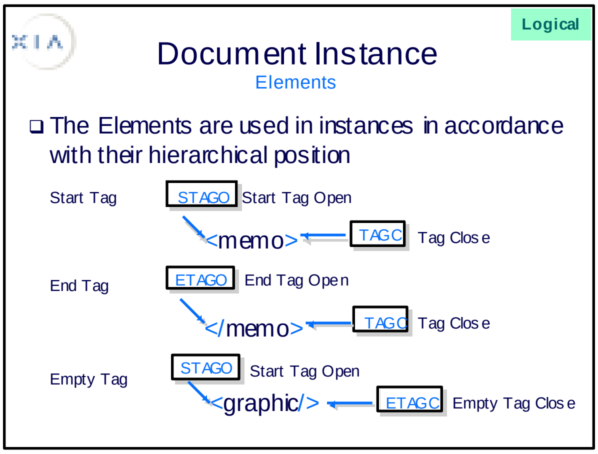
ANY

- A declared content of ANY (keyword)
  - ANY means that any Element declared in the DTD referenced by that DOCTYPE declaration is allowed inside the Element as well as #PCDATA

    <!ELEMENT doc (chap)>
    <!ELEMENT chap (title,p+)>
    <!ELEMENT title (#PCDATA)>
    <!ELEMENT p ANY>

    This declaration for p is equivalent to:

    <!ELEMENT p (#PCDATA | doc | chap | title | p)* >



Logical

Document Instance

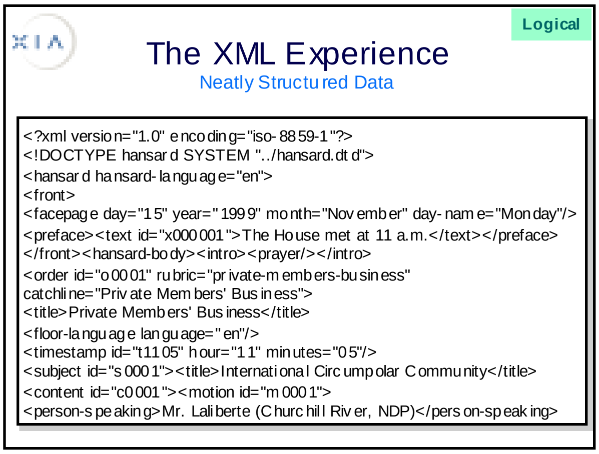
Elements

- The Elements are used in instances in accordance with their hierarchical position

Start Tag
STAGO Start Tag Open
<memo> TAGC Tag Close

End Tag
ETAGO End Tag Open
</memo> TAGC Tag Close

Empty Tag
STAGO Start Tag Open
<graphic/> ETAGC Empty Tag Close

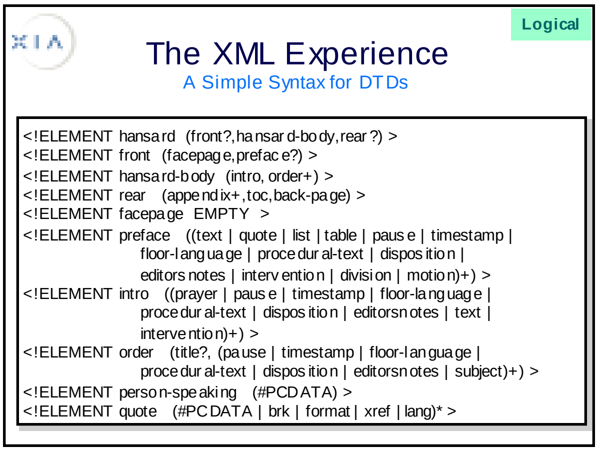


Logical

The XML Experience

Neatly Structured Data

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE hansard SYSTEM "../hansard.dtd">
<hansard hansard-language="en">
<front>
<facepage day="15" year="1999" month="November" day-name="Monday"/>
<preface><text id="x000001">The House met at 11 a.m.</text></preface>
</front><hansard-body><intro><prayer/></intro>
<order id="o0001" rubric="private-members-business"
catchline="Private Members' Business">
<title>Private Members' Business</title>
<floor-language language="en"/>
<timestamp id="t1105" hour="11" minutes="05"/>
<subject id="s0001"><title>International Circumpolar Community</title>
<content id="c0001"><motion id="m0001">
<person-speaking>Mr. Laliberte (Churchill River, NDP)</person-speaking>
```

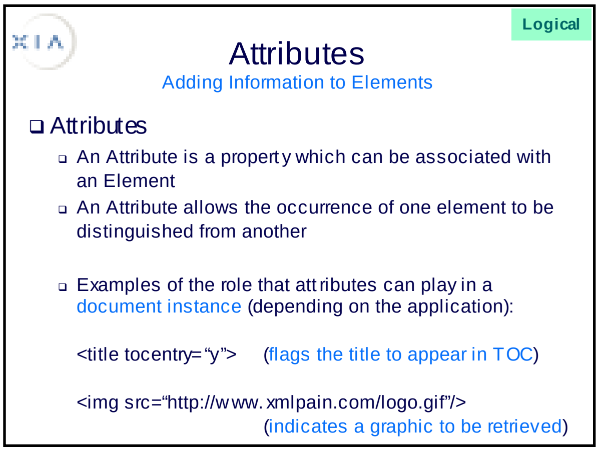


Logical

The XML Experience

A Simple Syntax for DTDs

```
<!ELEMENT hansard  (front?,hansard-body,rear?) >
<!ELEMENT front  (facepage,preface?) >
<!ELEMENT hansard-body  (intro, order+) >
<!ELEMENT rear  (appendix+,toc,back-page) >
<!ELEMENT facepage  EMPTY  >
<!ELEMENT preface   ((text | quote | list | table | pause | timestamp |
                    floor-language | procedural-text | disposition |
                    editorsnotes | intervention | division | motion)+) >
<!ELEMENT intro   ((prayer | pause | timestamp | floor-language |
                    procedural-text | disposition | editorsnotes | text |
                    intervention)+) >
<!ELEMENT order   (title?, (pause | timestamp | floor-language |
                    procedural-text | disposition | editorsnotes | subject)+) >
<!ELEMENT person-speaking   (#PCDATA) >
<!ELEMENT quote   (#PCDATA | brk | format | xref | lang)* >
```



Logical

Attributes

Adding Information to Elements

- Attributes
  - An Attribute is a property which can be associated with an Element
  - An Attribute allows the occurrence of one element to be distinguished from another
  - Examples of the role that attributes can play in a document instance (depending on the application):

    <title tocentry="y">   (flags the title to appear in TOC)

    <img src="http://www.xmlpain.com/logo.gif"/>
    (indicates a graphic to be retrieved)

Logical

Attributes

Typical Uses

❑ Attributes add information about elements that will be used by an application

<memo status="draft">
<artwork effectivity="modelA1">

❑ Attributes can indicate the functionality to be applied to an Element

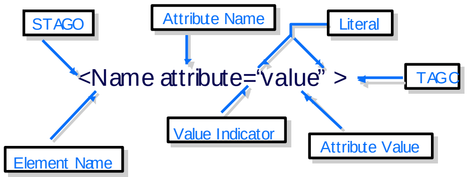
<emphasis style="italics">*these words*</emphasis>
<change type="delete">this phrase</change>

Logical

Attributes

Specifying Attribute Values in an Instance

❑ Attributes appear within a Document Instance *inside* the start tag of the Element it is qualifying



<Name attribute="value" >

<date day="fri" moon="full">13 Apr 92</date>

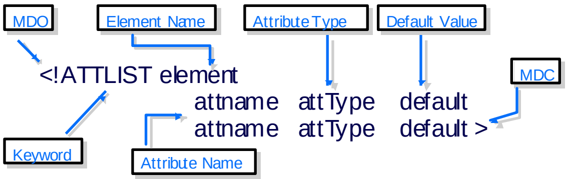
Logical

Attribute Declarations

❑ All Attributes must be declared

❑ Attributes are associated with an element with an attribute-list declaration

❑ The sequence within an ATTLIST is arbitrary



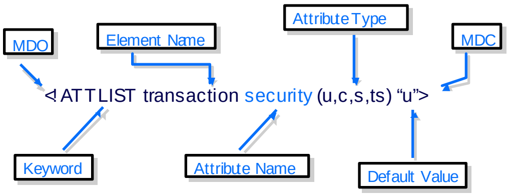
<!ATTLIST element
attname attType default
attname attType default >

Logical

Attribute Declarations

Example

❑ A declaration for a single attribute that is associated with the Element transaction:



<!ATTLIST transaction security (u,c,s,ts) "u">

Logical

Attribute Declarations

Default Values

❑ Default Value Keywords

- identifies what the parser should do if no value is declared
  - **#REQUIRED** - The Attribute **must** be specified. No action will be taken to default a value
  - **#IMPLIED** - The Attribute is optional. If no value is declared, the parser will not assume a value
  - **#FIXED** - The parser will always assume the fixed default value, regardless of what may be added to the instance

Logical

Attribute Declarations

Attribute Types

❑ XML allows for only certain attribute types

- CDATA - Character Data (string)
- NMTOKEN - single name token
- NMTOKENS - a series of name tokens
- ID - Identifier value (unique within an instance)
- IDREF - Identifier Reference
- IDREFS - series of Identifier References
- NOTATION - a selection from Notation Names
- ENUMERATION - a selection from group of NMTOKENS
- ENTITY - Entity reference
- ENTITIES - a series of Entity references

Logical

Attributes

Special Case: ID / IDREF

❑ Attributes can be used to identify an Element and then establish a reference to the Element

<figure ID="A10019039J">

<figref IDREF="A10019039J">

This relationship can become a hypertext link or a generated note as in (See Figure 6.1)

❑ The ID Value will be validated as unique within the document instance in which it occurs

❑ The IDREF Value will be validated as an existing ID Value on an Element within the same document instance

Logical

Attributes

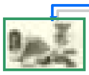
Special Case: Entity Reference

❑ Attributes can identify the name of an entity whose content is to be processed at the point where the entity reference is made

<graphic boardno="Z00909B" />

When the attribute type is declared as an Entity Reference, then the name included as the attribute value will be validated as a declared entity

<!ENTITY Z00909B SYSTEM "Z00909B.tif" NDATA tiff>



This is the most common way to reference unparsed external entities such as graphics or media segments

Physical

Entities

Definition

❑ Entities

- are used as shorthand in the DTD or in the Document Instance to reference separately stored units of data
- are used to reference separate files
  - XML data (parsed)
  - Non-XML data (unparsed)
- are used to enter special characters that cannot be keyed in on a keyboard

Physical

Entities

Definitions

❑ Entities are classified into

- Internal (entity value provided in the DTD)
- External (entity value is stored outside DTD)

❑ There are two types of Entities

- Parameter (used in the DTD)
- General (used in a Document Instance)

❑ Entities are also either

- Parsed (contents treated as replacement text)
- Unparsed (non-XML data handled as per a NOTATION)

Physical

Entity Definitions

Defining the content of an entity

❑ The content of an entity can be:

- an entity value provided in the entity declaration in the DTD (internal)

  In the DTD: <!ENTITY lh "Locking Hardware">
  In the Instance: <step>Remove &lh;</step>
  Becomes: <step>Remove Locking Hardware</step>

- an identifier that "points to" the entity content (external):

  SYSTEM Identifier:
  <!ENTITY logo SYSTEM "http://www.xiacorp.com/logo.gif">

  PUBLIC Identifier (always followed by SYSTEM identifier):
  <!ENTITY notice PUBLIC "-//XIACORP//TEXT Notice//EN"
  "http://www.xiacorp.com/notice.txt" >

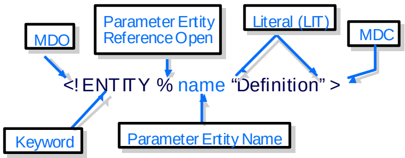
Physical

Parameter Entities

Definition

❑ Parameter Entities are declared and referenced in the DTD as a means to organize and manage markup declarations

- The Structure of a Parameter Entity Declaration is:


MDO
Parameter Entity Reference Open
Literal (LIT)
MDC
<!ENTITY % name "Definition" >
Keyword
Parameter Entity Name

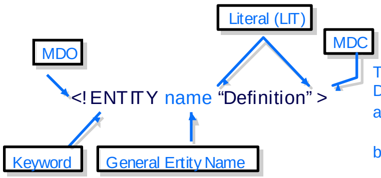
- Elsewhere in the DTD the entity is referenced as %name; which is replaced by the content of the entity definition

Physical

General Entity Declarations

Definition

- ❑ General Entities are declared in the DTD and referenced in the Document Instance
  - ❑ The Structure of a General Entity Declaration is



The content of the Entity Definition can be:

a. Provided internally in the Entity Declaration

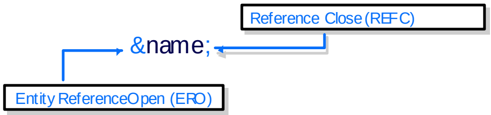
b. Referenced as external content using a System Identifier or a Public Identifier followed by a System Identifier

Physical

General Entity Reference

Pulling content into a document instance

- ❑ The General Entity is referenced in the Document Instance:



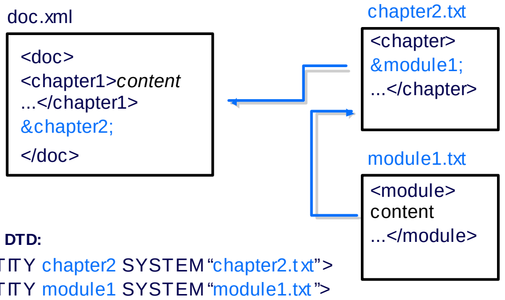
Examples:

Declared	Entered	Rendered
<!ENTITY corpabbr "xiacorp">	&corpabbr;	xiacorp
<!ENTITY cortact "www.&corpabbr;.com">	See &cortact;	See www.xiacorp.com

Physical

Parsed General Entities

Using External General Entities to reuse content



doc.xml

```
<doc>
<chapter1>content
...</chapter1>
&chapter2;
</doc>
```

chapter2.txt

```
<chapter>
&module1;
...</chapter>
```

module1.txt

```
<module>
content
...</module>
```

In the DTD:

<!ENTITY chapter2 SYSTEM "chapter2.txt">
<!ENTITY module1 SYSTEM "module1.txt">

Physical

Unparsed General Entities

Enabling Multimedia

- ❑ External General Entities can reference unparsed (non-XML) data
  - ❑ Graphics or Multimedia objects
- ❑ A NOTATION is used to declare basic processing information so it is available to the application attempting to process the Unparsed data

<!NOTATION TeX SYSTEM "../TeXView.exe">

Physical

Using Notations and Entities

Enabling Multimedia

- ❑ Example

```
<? XML version="1.0" encoding="UTF-8" ?>
<!DOCTYPE doc SYSTEM "doc.dtd" [
<!ENTITY fig1 SYSTEM "fig1.tif" NDATA tiff>  ]>
<doc>...
<graphic boardno="fig1"/>
</doc>
```

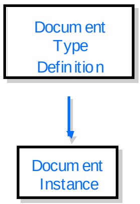
In the DTD:

```
<!NOTATION tiff SYSTEM "../rasterview.exe">
```

Physical

Document Instances

How they pull the pieces together



```
<!ENTITY Z000005-tif SYSTEM "Z000005a.tif" NDATA tiff>
<!NOTATION tiff SYSTEM "../rasterview.exe">
<!ELEMENT graphic  EMPTY >
<!ATTLIST graphic
          boardno  ENTITY                      #REQUIRED
          alignto (top,first,bottom,last)      #IMPLIED
          scalefit NMTOKEN                     #IMPLIED
          scalewd  NMTOKEN                     #IMPLIED  >

<?XML version="1.0"?>
<!DOCTYPE assemby SYSTEM"assembly.dtd" []>
<assembly>
<diagram><figure id="G000001-AA04">
<title>Suspension System Assembly</title>
<graphic alignto="top" boardno="Z000005-tif"
 scalefit="1"scalewd="504"/></figure>
</diagram>
</assembly>
```

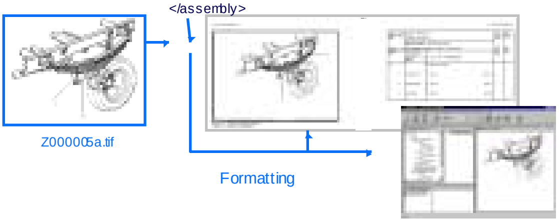
Physical

XML and other formats

Including unparsed entities

Docum ent Instance

```
<?XML version=“1.0”?>
<!DOCTYPE assembly SYSTEM”assembly.dtd”>
<assembly><diagram><figure id="G000001-AA04">
<title>Suspension System Assembly</title>
<graphic alignto=”top" boardno="Z000005-tif”
 scalefit="1"scalewd="504”/></figure></diagram>
</assembly>
```



Summary

- ❑ XML
  - ❑ A World Wide Web Consortium Recommendation
  - ❑ An application profile, or restricted form, of SGML, the Standard Generalized Markup Language
  - ❑ Provides the benefits of SGML to Web applications in a manner that is simple to implement
  - ❑ Is extensible (like SGML) and provides the mechanisms whereby users define their own markup languages

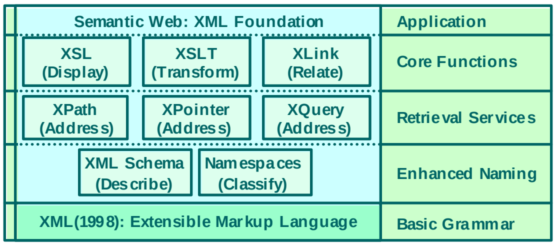
XML

The XML Family

of Companion Recommendations

The XML Family

XML Implementation Components



Semantic Web: XML Foundation | Application

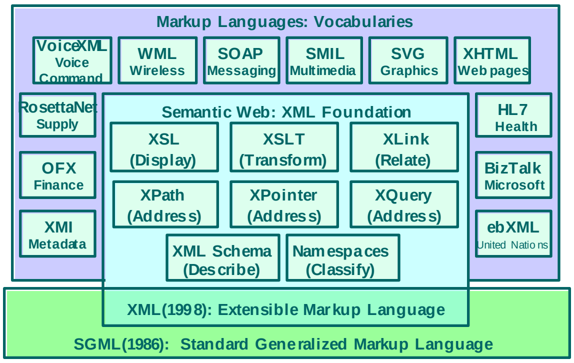
XSL (Display) | XSLT (Transform) | XLink (Relate) | Core Functions

XPath (Address) | XPointer (Address) | XQuery (Address) | Retrieval Services

XML Schema (Describe) | Namespaces (Classify) | Enhanced Naming

XML(1998): Extensible Markup Language | Basic Grammar

The XML Universe



Markup Languages: Vocabularies

VoiceXML Voice Command

WML Wireless

SOAP Messaging

SMIL Multimedia

SVG Graphics

XHTML Web pages

RosettaNet Supply

OFX Finance

XMI Metadata

HL7 Health

BizTalk Microsoft

ebXML United Nations

Semantic Web: XML Foundation

XSL (Display)

XSLT (Transform)

XLink (Relate)

XPath (Address)

XPointer (Address)

XQuery (Address)

XML Schema (Describe)

Namespaces (Classify)

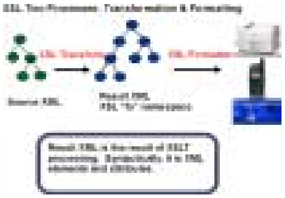
XML(1998): Extensible Markup Language

SGML(1986): Standard Generalized Markup Language

XSL

XML Implementation Components

- ❑ Extensible Stylesheet Language (XSL)
  - ❑ Working Draft dated October 18, 2000
  - ❑ An advanced language for expressing stylesheets
  - ❑ Stylesheets entail transformations (markup and data manipulation) and formatting semantics (rendition instructions)



```
<xsl template match="name">
<p style="fort-size: smaller;
    margin-top: 0; margin-bottom: 0">
 <a target="summary">
    <xsl:attribute name="href">
    <xsl:value-of select="@xlink:href"/>
    </xsl:attribute>
  <xsl:value-of select="@xlink:ttle"/>
 </a>
</p>
</xsl:template>
```

Example: W3C XSL Spec

XSLT

XML Implementation Components

❑ XSL Transformations (XSLT)

- W3C Recommendation dated November 16, 1999
- The specification defining the language for transforming XML Documents into other XML Documents
- A subset of XSL that defines transformations using well-formed XML documents

```
<xsl:template match="facepage">
          <xsl:if test="/toc/@language[.='en']">
                    <p class="toc_contents">CONTENTS</p>
                    <p class="toc_date">
                    <xsl:value-of select="@day-name"/>,
                    <xsl:value-of select="@month"/>
                    <xsl:value-of select="@day"/>,
                    <xsl:value-of select="@year"/>    </p>...
```

XLink

XML Implementation Components

❑ XML Linking Language (XLink)

- Candidate Recommendation July 3, 2000
- XLink allows elements to be inserted into documents to describe relationships between resources and sub-resources
- Replacing the humble uni-directional anchor tag <a href=“”></a>
- XLinks will implement:
  - bi-directional links
  - link metadata
  - link behaviour
  - out-of-line links that can be managed independently

```
<my:crossReference
  xmlns:my="http://example.com/"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xlink:type="simple"
  xlink:href="students.xml"
  xlink:title="Student List"
  xlink:actuate="onRequest">
Current List of Students
</my:crossReference>
```

XPath

XML Implementation Components

❑ XML Path Language (XPath)

- W3C Recommendation dated November 16, 1999
- A foundation language for addressing parts of an XML Document
- Developed for common use within both XSLT and XPointer
- XPath uses a compact non-XML syntax suitable for use within URIs and XML Attribute Values

Provides mechanism to identify a specific document component through a location path and a boolean or value based test

```
child::*    selects all element children of the context node
child::para[position()=last()]    selects the last para child of the context node
child::para[position()=last()-1]    selects the last but one para child
                                    of the context node
```

XPointer

XML Implementation Components

❑ XML Pointer Language (XPointer)

- Candidate Recommendation dated June 7, 2000
- An extension to the Uniform Resource Identifier (URI) reference
- Extends referencing into “sub-resources” (e.g. fragments)
- To provide an addressing method that operates without a need for a physical target identifier existing on the sub-resource
- Based and Extends the XPath Recommendation adding:
  – points and ranges to be addressed as well as nodes
  – information location using string matching
  – addressing expressions to be used in URIs as fragment identifiers

```
string-range(//title,”Yuri Rubinsky")[17]

Returns the 17th occurrence of “Yuri Rubinsky” within a Title Element
```

XQuery

XML Implementation Components

❑ XML Query language

- Requirements Working Draft August 15, 2000
- Data Model Working Draft May 11, 2000
- The objective is the development of a data model for XML Documents, a set of query operators, and a query language based on those operators.
- Sets of XML Documents would be queried like a database
- Results would be extracted and returned for processing
- Queries could be issued against relational databases with results returned in XML for processing

XML Schema

XML Implementation Components

❑ XML Schema (Structures and Datatypes)

- Working Draft Status April 7, 2000
- Schema Structures (Part 1) defines an “instance-based” method for describing document structures and content
- Schema Datatypes (Part 2) defines an “instance-based” method for defining the datatypes that can exist in XML documents

```
<ElementType name= "primaryKey" content = "eltOnly" order = "seq">
   <AttributeType  name =  "name" dt:type=  "string"  required = "yes"/>
   <AttributeType  name =  "linkName" dt:type =  "id"  required = "yes"/>
   <AttributeType  name =  "enabled"  dt:type =  "enumeration"  dt:values=  "yes no"
                required = "yes"/>
   <attribute type =  "name"/>
   <attribute type =  "linkName"/>
   <attribute type =  "enabled"/>
   <element type = "columnLink" minOccurs = "1" maxOccurs=  "*"/>
</ElementType>
```

Namespaces

XML Implementation Components

- ❑ XML Namespaces
  - A W3C Recommendation dated January 14, 1999
  - "An XML namespace is a collection of names, identified by a URI reference, which are used in XML documents as element types and attribute names" W3C
  - Intended to allow data from different sources, which may have identical names but follow different rules, to be mixed together
  - Allows element instances to declare their "parent" markup languages (vocabularies)

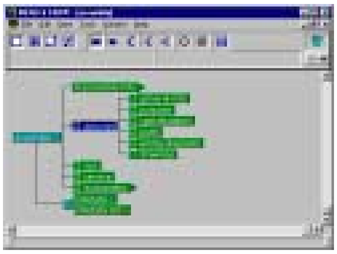
```
<x xmlns:edi='http://ecommerce.org/schema'>
<edi:price units='Euro'>32.18</edi:price></x>
```

The namespace for the element price is http://ecommerce.org/schema

XML
Available Technology

DTD Creation Tools

- ❑ Near & Far

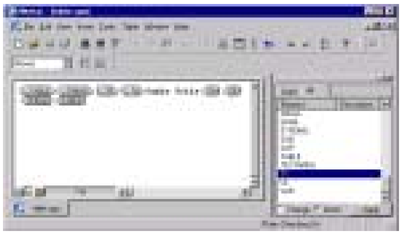


DTD Creation Tools

- ❑ Applications designed to help DTD Developers
- ❑ Offer a graphical view of a DTD and provide analytical reports (e.g. attribute usage)
- ❑ Examples:
  - Near & Far - OpenText Corporation
    - View, create and edit DTDs graphically
    - XML Version
    - XML / SGML Version
  - XML Authority - Extensibility
  - Schema Central - XML Solutions
    - Schema Management and Conversion
  - Document Architect - ArborText Inc.

XML Editors

- ❑ XMetal

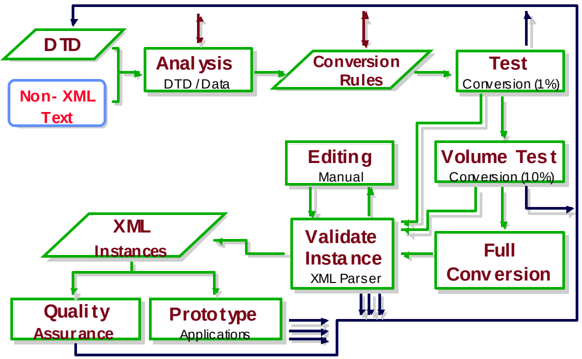


XML Editors

- ❑ Examples
  - XMetaL 2.0 - SoftQuad
  - EPIC (ADEPT Editor) - ArborText
  - Documentor 2.4 - Exosoft
  - EditTime 3.0 - TimeLux
  - XMLSpy 3.0 - Icon Information Systems
  - Xeena 1.2 - IBM Alphaworks
  - Morphon (Beta) - Lunatech Research
  - XML Notepad - Microsoft
  - GRIF Editor - Infrastructures for Information (I4I)
  - QuickSilver - Broadvision (Interleaf)
  - XML Pro - Vervet Logic

Conversion to XML

Adding Intelligence to Documents



XML Conversion Tools

- Examples
  - Omni Mark - Omni Mark
  - Balise - AIS Software (eBT)
    - Note: Inso became Electronic Business Technologies (eBT)
  - DynaTag - Enigma
  - Rainbow Maker (RTF to Markup) - Public Domain
  - FastTag - OpenText (not supported)
  - IntelliTag - Corel WordPerfect (not supported)

Parsing and Validating XML

SGML Document
test.xml

```
<!DOCTYPE test [
<!ELEMENT test (A,B,C)>
<!ELEMENT A (#PCDATA)>
<!ELEMENT B (#PCDATA)>
<!ELEMENT C (#PCDATA)>
 ]>
<test>
<A>Content</A>
<C>Content</C>
<B>Content</B>
</test>
```

Parser Output
OmniMark

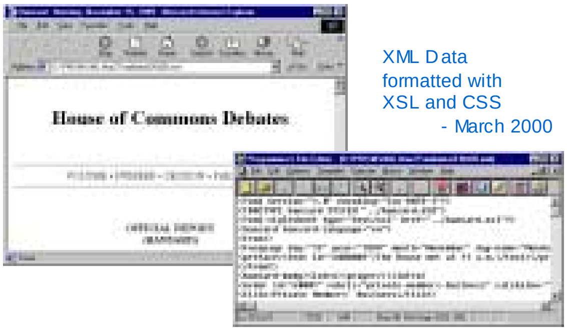
```
OmniMark 4.0.1
Copyright, (C) 1988-99 OmniMark
Technologies Corporation
Warning on line 6 in file
test.xml
A start tag is not allowed at the
current point.
The element is “C.”
The open element is “TEST”.
The following element can start: “B”
```

Parsers

- Examples
  - Omni Mark - Omni Mark Technologies
  - Mark-it - Sema Group
  - NSGMLS / SP / XP - Public Domain (James Clark)
  - Microsoft XML Parser - Microsoft
  - JAXP (Java) - Sun Microsystems
  - Oracle XML Parser (Oracle 8i r3) - Oracle
  - Xerces (validating) - Apache XML Project
  - IBM XML Parser for C (based on Xerces) - IBM
  - IBM XML Parser for Java (validating) - IBM
  - Many more: mostly non-validating

XML Publishing

Even on the Web!



House of Commons Debates

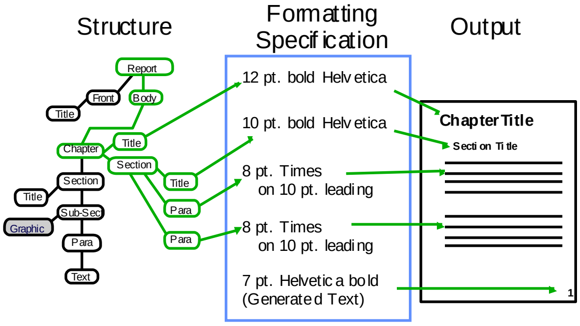
OFFICIAL REPORT (HANSARD)

XML Data formatted with XSL and CSS

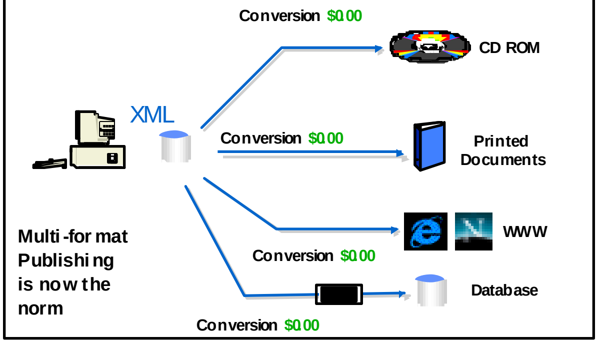
- March 2000

XML Publishing

Mapping Formatting to Structure



Multi-format Publishing



Conversion $0.00

CD ROM

XML

Conversion $0.00

Printed Documents

WWW

Conversion $0.00

Database

Conversion $0.00

Multi-format Publishing is now the norm

XML Repositories

Classical

- ❑ Examples:
  - BladeRunner/Information Manager-Broadvision (Interleaf)
  - Astoria - Chrystal Software
  - Parlance Content Management - XyEnterprise
  - Poet Content Manager - Poet Software
  - Tamino - Software AG
  - WorkSMART - OpenText Corporation
  - Engenda / DynaBase - eBT
  - V/5 Content Management Server - Vignette
  - 4i Content Management - Documentum
  - Expressroom - Worldweb.net
  - XML Canon - Extensibility (Beta)

XML Repositories

Other Models

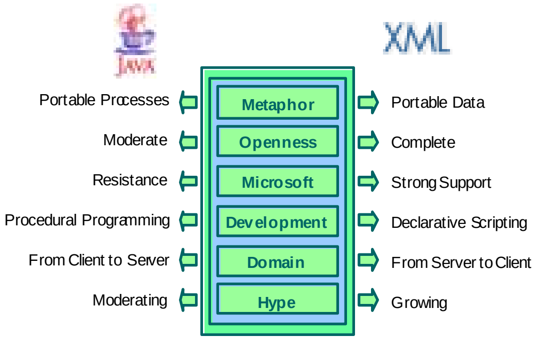
- ❑ XML in SQL Databases:
  - LivePage - Janna (SQL database)
  - XBase - Eidon (SQL database)
  - Oracle 8i r3 - Oracle
  - SQL Server 2000 - Microsoft
- ❑ Text Databases
  - BASISplus - OpenText Corporation
  - TextML - IXIASOFT
  - SIM - Aspect Computing

Role

Where does XML fit?

XML and Java

Made for Each Other



Portable Processes	Metaphor	Portable Data
Moderate	Openness	Complete
Resistance	Microsoft	Strong Support
Procedural Programming	Development	Declarative Scripting
From Client to Server	Domain	From Server to Client
Moderating	Hype	Growing

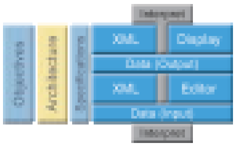
Modified from: Gartner Group 2000

The XML Data Type

Its just a data type!

XML does not do anything

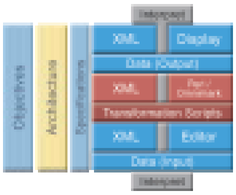
It can be always read...



The XML Data Type

XML must always be interpreted

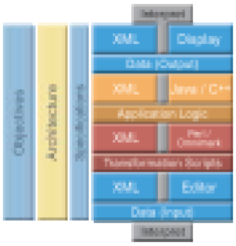
XML must always be processed



The XML Data Type

Application Logic can be applied to XML

XML can also be used to describe or declare parameters for application logic

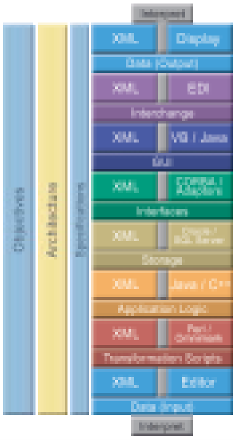


The XML Data Type

In complex systems XML can play a role at each level

The Modern Reality: Complex Systems are always loosely coupled

XML:
The WD40 or
Duct Tape of
System Integration

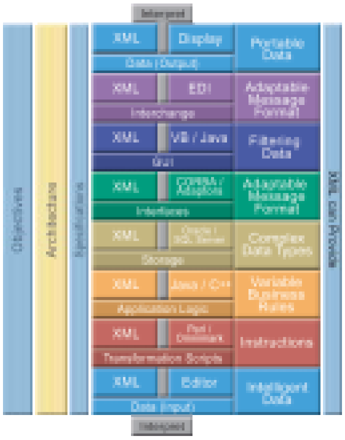


The XML Data Type

XML is the last Data Type

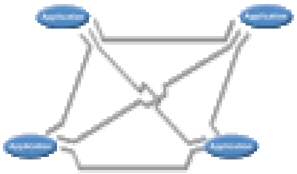
It is extensible and based on human communication structures

There is data that can only be described in XML



XML in Practice

One Piece in a Larger Puzzle



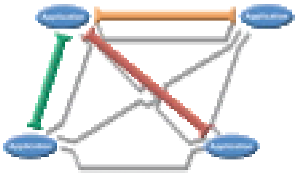
XML Messages can appear crude but they will always work

XML can be used to define the message format for all information exchanges

This requires that each node has an interpretation ability

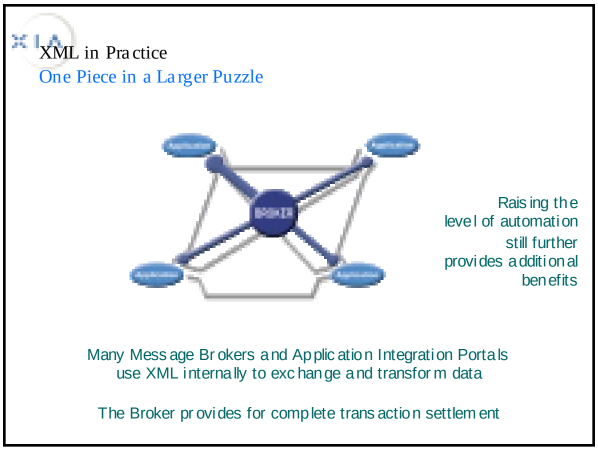
XML in Practice

One Piece in a Larger Puzzle



Raising the level of automation being applied reduces the recurrent work load

For known and repeated interchanges Applications Components are implemented to handle the full transaction



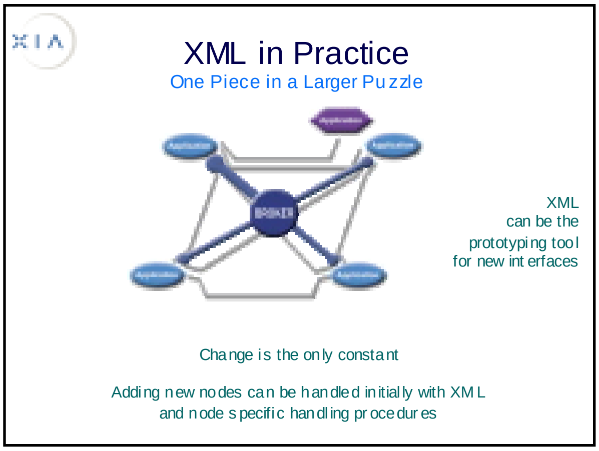
XML in Practice

One Piece in a Larger Puzzle

Raising the level of automation still further provides additional benefits

Many Message Brokers and Application Integration Portals use XML internally to exchange and transform data

The Broker provides for complete transaction settlement



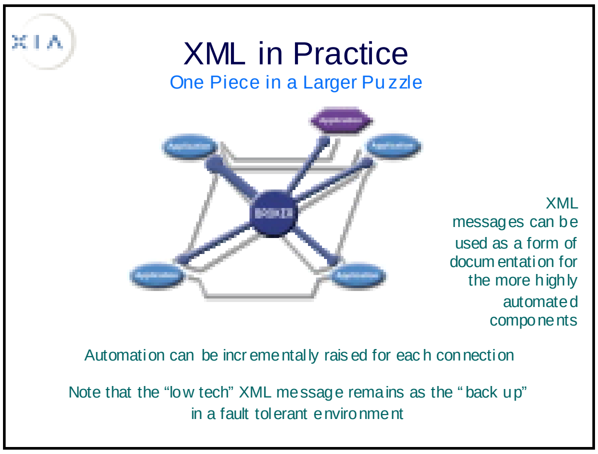
XML in Practice

One Piece in a Larger Puzzle

XML can be the prototyping tool for new interfaces

Change is the only constant

Adding new nodes can be handled initially with XML and node specific handling procedures



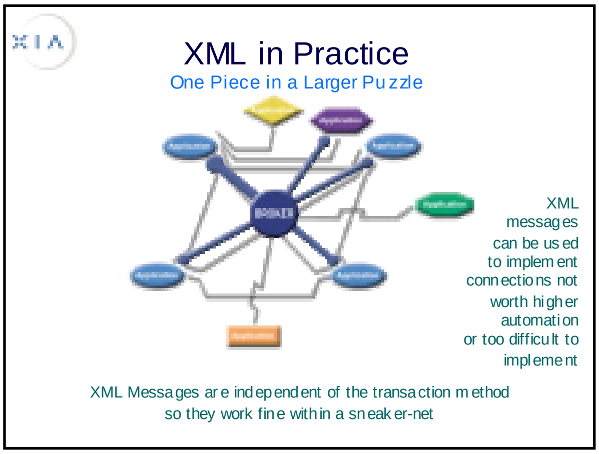
XML in Practice

One Piece in a Larger Puzzle

XML messages can be used as a form of documentation for the more highly automated components

Automation can be incrementally raised for each connection

Note that the "low tech" XML message remains as the "back up" in a fault tolerant environment

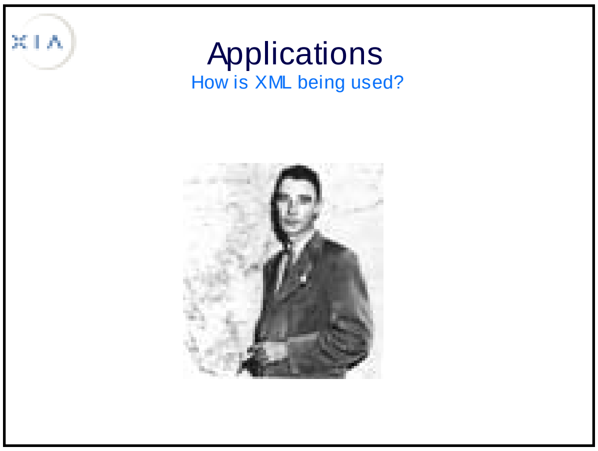


XML in Practice

One Piece in a Larger Puzzle

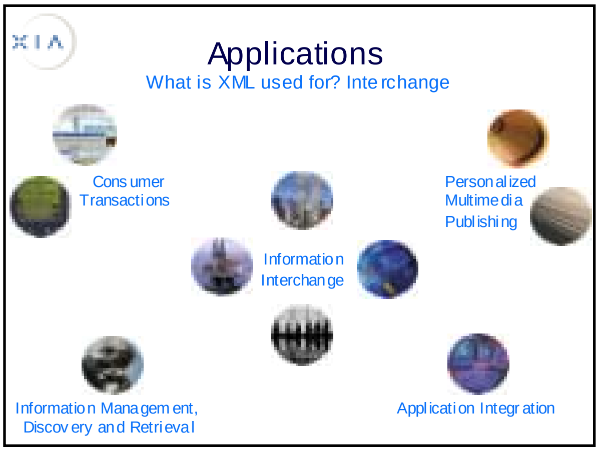
XML messages can be used to implement connections not worth higher automation or too difficult to implement

XML Messages are independent of the transaction method so they work fine within a sneaker-net



Applications

How is XML being used?



Applications

What is XML used for? Interchange

Consumer Transactions

Personalized Multimedia Publishing

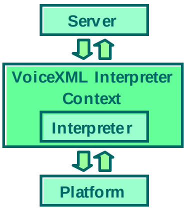
Information Interchange

Information Management, Discovery and Retrieval

Application Integration

Voice XML

- ❑ Voice Extensible Markup Language
  - Initiative led by IBM, AT&T, Lucent, and Motorola
  - Designed for creating audio dialogs with synthesized speech and speech recognition
  - web enabling interactive voice response applications



```
<?XML version="1.0"?>
<vxml version="1.0">
<form><field name="drink">
<prompt>Coffee or Tea?</prompt>
<grammar src="drink.gram"
              type="application/x-jsqf"/></field>
<block><submit next="http://www.drink.com" />
</block></form>
</vxml>
```

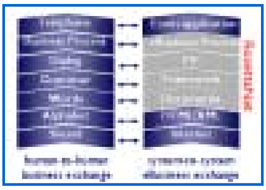
WML

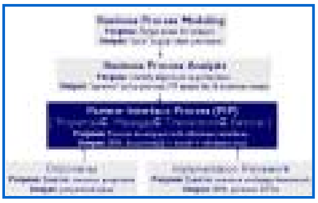
- ❑ Wireless Markup Language
  - Component of the Wireless Application Protocol (WAP)
  - Markup Language designed to support
    - data exchange and rendering in wireless applications
    - small bandwidth transmission requirements
    - interaction descriptions for implementation of different devices

```
<?XML version="1.0"?>
<wml>
<card id="abc" ordered="true">
<p><do type="accept"><go href="http://www.xyz.org" /></do>
        X: $(X)<br/>
        Y: $(&#59:)<br/>
        Enter Name: <input type="text" name="N" />
</p></card></wml>
```

RosettaNet

- ❑ Technology Supply Chain Integration
  - similar to, and subsuming, the Pinnacles Initiative in the semi-conductor industry
  - Provides a methodology for building interchange solutions


RosettaNet
human-to-human business exchange
system-to-system eBusiness exchange


Business Process Modeling
Business Process Analysis
Partner Interface Process (PIP)
Dictionaries
Implementation Framework

SVG

- ❑ Scalable Vector Graphics
  - W3C Candidate Recommendation November 2, 2000
  - A language for describing 2-dimensional graphics in XML
    - vector graphic shapes, images and text (XML)
    - graphics animation

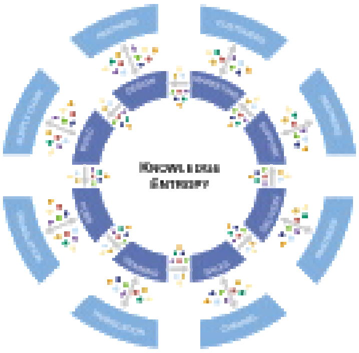
```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg SYSTEM "http://www.w3.org/TR/2000/CR-SVG-20001102/DTD/svg-20001102.dtd">
<svg width="5cm"  height="5cm">
  <desc>Two groups, each of two rectangles</desc>
  <g id="group1" style="fill:red">
    <rect x="1cm" y="1cm" width="1cm" height="1cm" />
    <rect x="3cm" y="1cm" width="1cm" height="1cm" />   </g>
</svg>
```

ICE

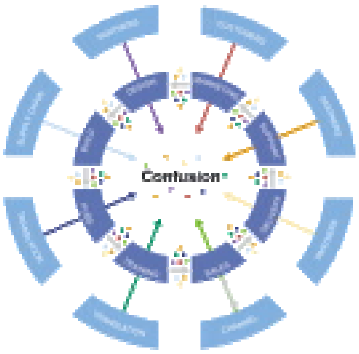
- ❑ Information and Content Exchange
  - Creates a common language and protocol for the automatic exchange of content assets
  - Integrating web assets from the perspective of users:
    - Individual and corporate consumers
    - Syndication Service Providers (Web Super stores)
    - Content Developers and Owners
  - Designed around a set of specific transactions were content is either sold, resold or licensed.

```
<ice-response request-id="1998-07-05T02:03:45@nr3.com-1" >
<ice-code numeric="200" phrase="OK" message-id="1998-08-11T12:34:56@xyz.com-1" >
</ice-code> </ice-response>
```

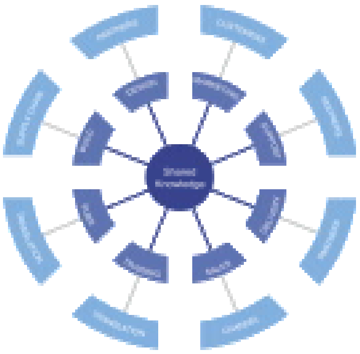
Knowledge Management



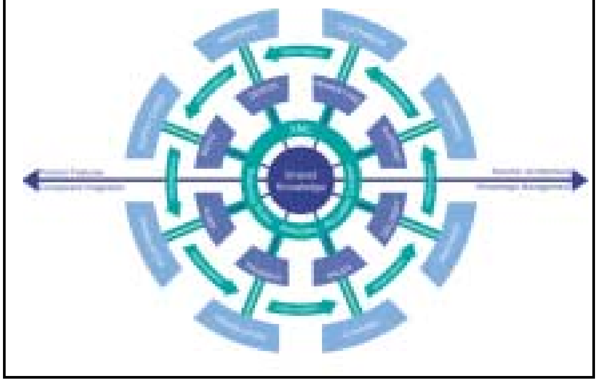
Knowledge Management



Knowledge Management



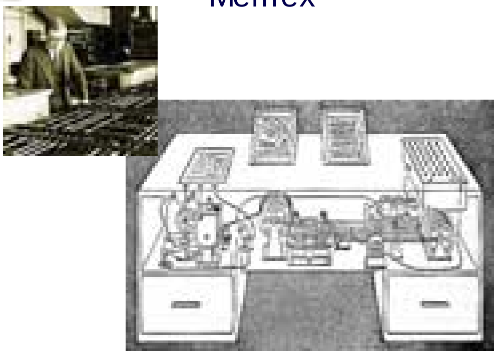
Knowledge Management



Conclusion

Where did XML come from and where is it going?

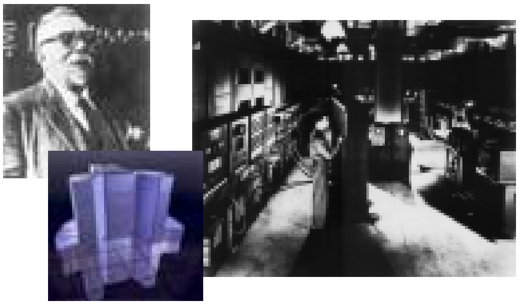
Memex



194 0 196 0 198 0 200 0

Control Systems

Applying Automation



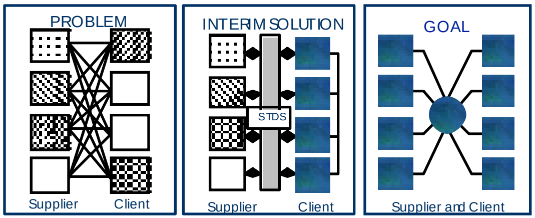
194 0 196 0 198 0 200 0

Internet

Connecting Organizations

CALS

US DOD tackles the problem head-on (1985)

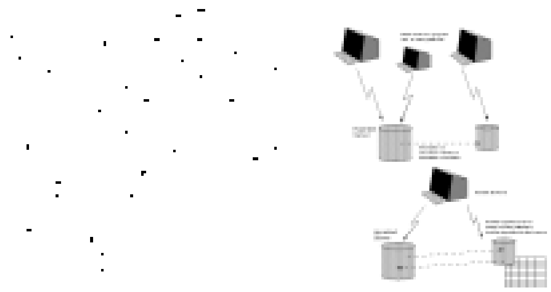


SGML

Standard Generalized Markup Language

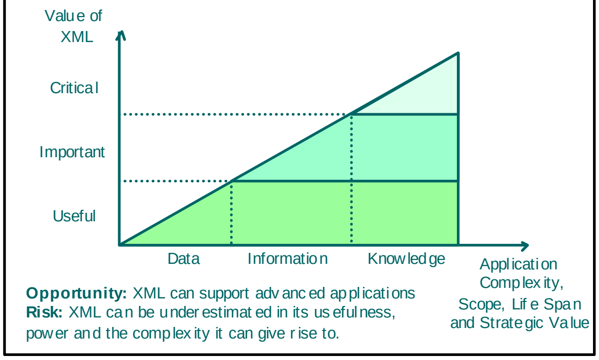
World Wide Web

Where there's a Will there's a Way

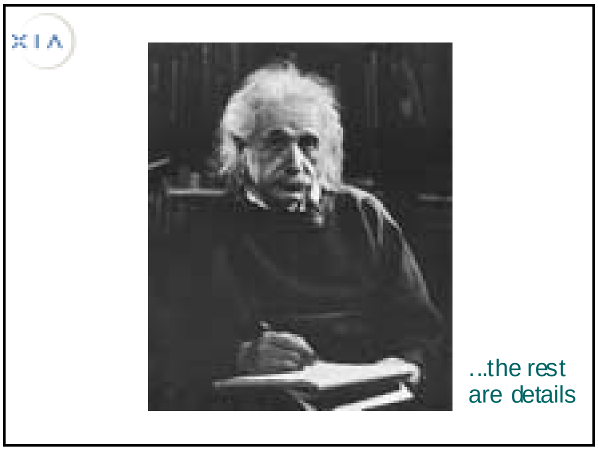


XML

XML



Opportunity: XML can support advanced applications
Risk: XML can be underestimated in its usefulness, power and the complexity it can give rise to.



...the rest are details

XIA

[illegible]

www.xiacorp.com