XML Seminar

W3C - WGI - CWI

World Wide Web Consortium
Werkgemeenschap Informatiewetenschap
Centrum voor Wiskunde en Informatica

Organization

World Wide Web Consortium (W3C)

- www.w3.org

Vereniging Werkgemeenschap Informatiewetenschap (WGI)

- www.cwi.nl/~lynda/WGI

Centrum voor Wiskunde en Informatica (CWI)

- www.cwi.nl

Program

Morning session

- Introduction to XML (Jacco van Ossenbruggen, CWI)
- XML and Style Sheets (Bert Bos, W3C, Editor of the CSS Recommendation)

Lunch (12.30 - 13.30)

Afternoon session (13.30 - 16:00)

- XML and HTML (Steven Pemberton, CWI, Chair of the HTML Working Group)
- Industrial applications of XML/SGML
  - Wim van Halteren (Elsevier, Director Electronic Products)
  - Frank van Harmelen (AIdministrator Nederland BV)

Borrel (16.00 - ...)

Introduction to XML

Jacco van Ossenbruggen

Multimedia and Human-Computer Interaction Group
CWI: Centrum voor Wiskunde en Informatica
Amsterdam, The Netherlands

Goals

Historical perspective:

- Explain who developed XML and why

Applications

- Sketch when you need XML and why

Concepts and syntax

- Demystify XML jargon
- Show example XML document and DTD

Future Directions

- Where the XML family is going
- Upcoming specifications
- What is still missing

Overview

Introduction

- What it is and what it is not
- Why do we need XML? We already have HTML!
- Why do we need XML? We already have SGML!

When do I need XML? (or: when are HTML and SGML not enough?)

- Typical XML applications
- XML processing

Concepts and Syntax: XML by Example

The rest of the XML family

Future directions

What is XML?

First, XML is NOT:

- a markup language!
- replacing HTML or SGML
- replacing programming or script languages such as Java or Javascript
- the silver bullet that solves all your IT problems
- just yet another overhyped internet buzzword

XML stands for eXtensible Markup Language

- a “meta” language which can be used to define a markup language

XML is developed by W3C

- to support dissemination of structured documents over the Web
- to overcome limitations of both HTML and SGML

Why do we need XML? We already have HTML!

Advantages of HTML:

- ubiquitous and portable
- software is cheap or free
- simple and easy to learn
- good language for simple, on-line hypertext documents

Disadvantages:

- fixed set of tags and attributes
- hard to tailor or extend
- hard to reuse:
  — for different media
  — in different contexts
  — for different users
- not suitable for (machine readable) semantic markup
- hard to implement due to backward compatibility requirements

Why do we need XML? We already have SGML!

Advantages of SGML:

- addresses all problems related to HTML
  - support for tailorable and extensible document markup
  - information can be reused because of abstractions provided by SGML
  - multiple delivery publishing (platforms, contexts, users)
  - supports semantic markup tailored to a specific application
- stable, platform and vendor neutral ISO standard since 1986

Disadvantages:

- complicated (not even fully implemented)
- expensive tools
- significant overhead
  - keystroke-level optimization
  - mandatory DTD
- ... and SGML is definitely not sexy!

XML sits right in the middle

XML inherits the tailorability and extensibility of SGML

- no fixed tag sets
- XML documents are tailorable and extensible
- multiple delivery publishing, semantic markup

By only using a minimal subset of SGML,
XML retains much of the advantages of HTML:

- XML is ubiquitous and portable (browsers, languages, operating systems)
- XML software is cheap or free
- XML is simple and easy to learn
- Less overhead: No keystroke-optimization, DTD is optional

Typical XML Applications

More complex documents that go beyond HTML ("SGML on the Web"):

- scientific publishing
- technical manuals

Non-text formats:

- multimedia (SMIL)
- vector graphics (SVG)

Database publishing and electronic publishing

- web site provides automated views on databases
- electronic data interchange (EDI) over the Web

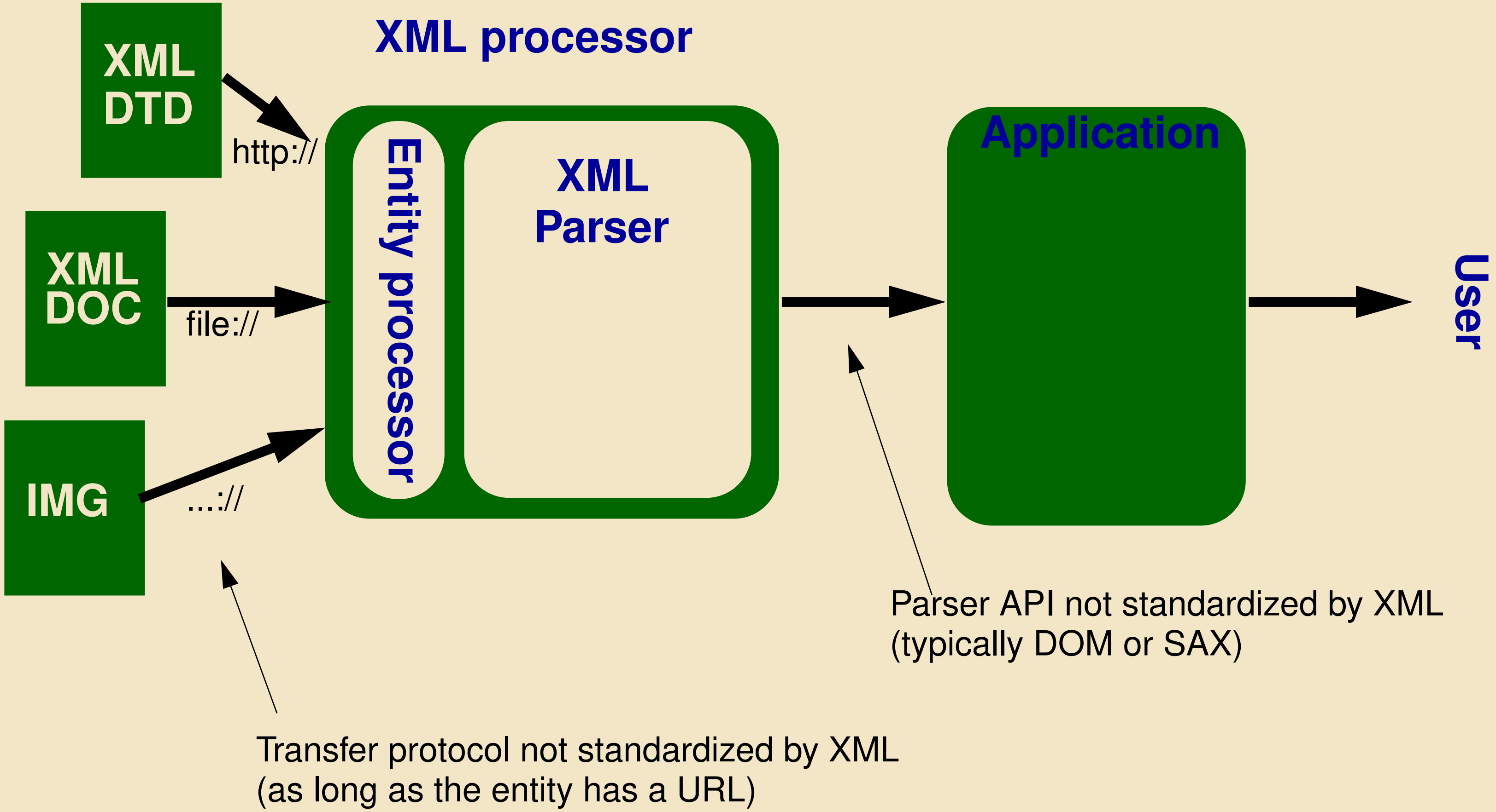
Meta data

- describing channels (Microsoft's CDF), Web sites (Netscape's MCF)
- filtering and content selection (W3C's PICS)
- general meta data syntax on top of XML (W3C's RDF)

But also: YOPLML

- Your Own Little Private Markup Language

XML processing pipeline



XML by Example: Our group's minutes

Group minutes of June 1, 1999

Participants: Lynda, Lloyd, Jacco
Regrets: Dick

Agenda

1) Last month's minutes
2) ...

...

Lynda asks whether this is a good XML example:
<date>June 1, 1999</date>

Action items

High priority:

Jacco: Prepare slides for the XML Seminar

...

XML by Example: content vs. markup

```
<?xml version="1.0"?>
<!DOCTYPE minutes SYSTEM "minutes.dtd">

<minutes>
  <date>June 1, 1999</date>
  <participants>Lynda, Lloyd, Jacco</participants>
  <regrets>Dick</regrets>
  <agenda>
   <item>Last month's minutes</item>
   ...
  </agenda>
  <issue id="i1">
    Lynda askes whether this is a good XML example:
    <![CDATA[<date>June 1, 1999</date>]]>
    ...
  <action name="Jacco" priority="high">
    Prepare slides for the XML Seminar
  </action>
  ...
</minutes>
```

XML by Example: tags

```
<?xml version="1.0"?>
<!DOCTYPE minutes SYSTEM "minutes.dtd">

<minutes>
  <date>June 1, 1999</date>
  <participants>Lynda, Lloyd, Jacco</participants>
  <regrets>Dick</regrets>
  <agenda>
   <item>Last month's minutes</item>
   ...
  </agenda>
  <issue id="i1">
    Lynda askes whether this is a good XML example:
    <![CDATA[<date>June 1, 1999</date>]]>
    ...
  <action name="Jacco" priority="high">
    Prepare slides for the XML Seminar
  </action>
  ...
</minutes>
```

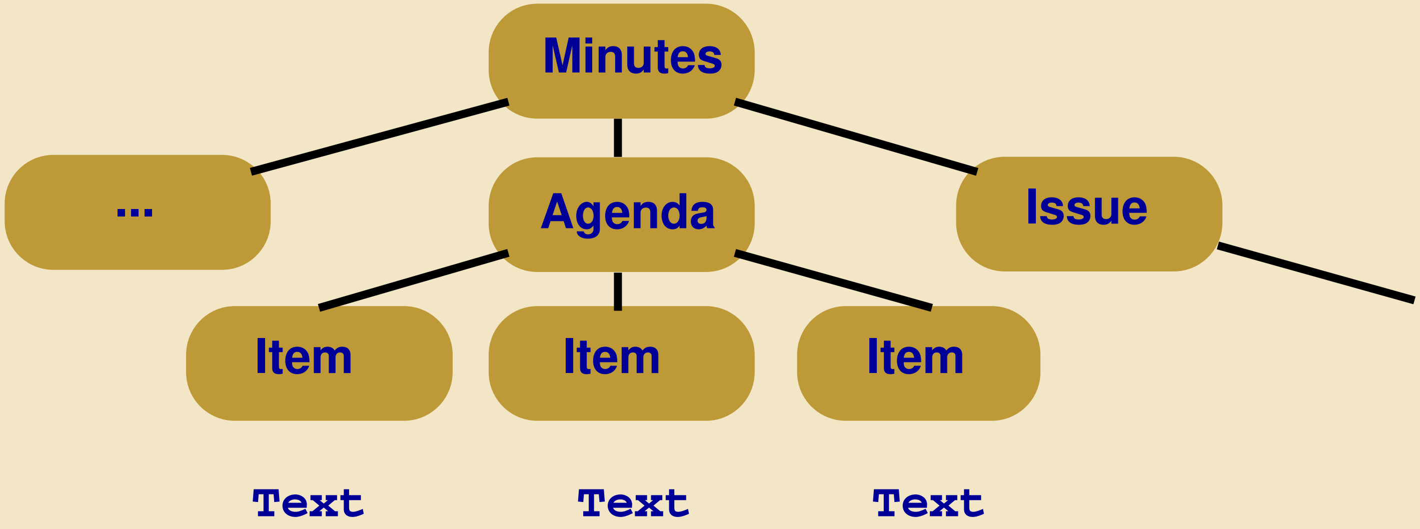
XML by Example: attributes

```
<?xml version="1.0"?>
<!DOCTYPE minutes SYSTEM "minutes.dtd">

<minutes>
  <date>June 1, 1999</date>
  <participants>Lynda, Lloyd, Jacco</participants>
  <regrets>Dick</regrets>
  <agenda>
   <item>Last month's minutes</item>
   ...
  </agenda>
  <issue id="i1">
    Lynda askes whether this is a good XML example:
    <![CDATA[<date>June 1, 1999</date>]]>
    ...
  <action name="Jacco" priority="high">
    Prepare slides for the XML Seminar
  </action>
  ...
</minutes>
```

XML by Example: elements

XML elements form the logical tree structure of each document:



Thus this is never allowed in XML:

```
<b>Some bold text <i> italic and bold </b> italic </i>
```

XML by Example: CDATA sections

```
<?xml version="1.0"?>
<!DOCTYPE minutes SYSTEM "minutes.dtd">

<minutes>
  <date>June 1, 1999</date>
  <participants>Lynda, Lloyd, Jacco</participants>
  <regrets>Dick</regrets>
  <agenda>
   <item>Last month's minutes</item>
   ...
  </agenda>
  <issue id="i1">
    Lynda askes whether this is a good XML example:
    <![CDATA[<date>June 1, 1999</date>]]>
    ...
  <action name="Jacco" priority="high">
    Prepare slides for the XML Seminar
  </action>
  ...
</minutes>
```

XML by Example: Processing instructions

```
<?xml version="1.0"?>
<!DOCTYPE minutes SYSTEM "minutes.dtd">

<minutes>
  <date>June 1, 1999</date>
  <participants>Lynda, Lloyd, Jacco</participants>
  <regrets>Dick</regrets>
  <agenda>
   <item>Last month's minutes</item>
   ...
  </agenda>
  <issue id="i1">
    Lynda askes whether this is a good XML example:
    <![CDATA[<date>June 1, 1999</date>]]>
    ...
  <action name="Jacco" priority="high">
    Prepare slides for the XML Seminar
  </action>
  ...
</minutes>
```

XML by Example: document type declaration

```
<?xml version="1.0"?>
<!DOCTYPE minutes SYSTEM "minutes.dtd">

<minutes>
  <date>June 1, 1999</date>
  <participants>Lynda, Lloyd, Jacco</participants>
  <regrets>Dick</regrets>
  <agenda>
   <item>Last month's minutes</item>
   ...
  </agenda>
  <issue id="i1">
    Lynda askes whether this is a good XML example:
    <![CDATA[<date>June 1, 1999</date>]]>
    ...
  <action name="Jacco" priority="high">
    Prepare slides for the XML Seminar
  </action>
  ...
</minutes>
```

XML by Example: Minutes DTD

```
<!-- This is the minutes DTD (optional) -->
<!ELEMENT minutes (date,participants,regrets?,
                        agenda,issue*,action*)>
<!ELEMENT agenda (item+)>
<!ELEMENT date         (#PCDATA)>
<!ELEMENT participants (#PCDATA)>
<!ELEMENT regrets      (#PCDATA)>
<!ELEMENT issue        (#PCDATA)>
<!ELEMENT action       (#PCDATA)>
<!ELEMENT item         (#PCDATA)>
<!ATTLIST issue
          ID       ID                 #IMPLIED>
<!ATTLIST action
          name     CDATA              #REQUIRED
          priority (high|medium|low)  “low”
>

<!-- “?” means optional
     “+” means one or more
     “*” means zero or more
-->
```

XML logical tree structure revisited

Logical structure

- content: plain text
- elements, tags and attributes (basic document structure)
- processing instructions (hints to your application)
- CDATA sections (content that should not be parsed)
- comments
- document type definition (DTD)

XML physical tree structure

Defines physical distribution of the parts that make up the document

- main document itself
- references to images, videos, ...
- the DTD
- a style sheet
- other XML fragments (banners etc)
- ...

and how these are accessed:

- files on your local file system
- web pages, images etc. via HTTP
- data via an interface to your database
- ... anything that has an URI (Uniform Resource Identifier)

XML Entities

These parts are called entities

- entities (also) form a tree structure
- you have at least one entity : the *document entity* at the root of the tree
- entity is referenced by using `&name;` syntax

Entities come in different flavors

- internal (defined within the document)
- external (defined elsewhere)
- unparsed (for binary data such as images)
- parsed (for inclusion of XML fragments)
- character (refers to character's (hexa)decimal number)
- parameter (only used in DTD)

XML Entities

Predefined entities		Character entities
<	<	<
>	>	>
&	&	&
‘	'	'
“	"	"

Example entity declarations/references

```
<!ENTITY XML “the extensible markup language”>
<!ENTITY Chapter1 SYSTEM “chapter-one.xml”>
<!ENTITY % HTMLlat1 PUBLIC
   "-//W3C//ENTITIES Latin1//EN//HTML"
   "HTMLlat1x.ent">
%HTMLlat1;
...
```

```
<!DOCTYPE book SYSTEM “book.dtd” [
<!ENTITY image1
         SYSTEM “http://www.site.com/img.gif” NDATA GIF>
]>
<book>
&copy; Jacco van Ossenbruggen
This is my book on &XML;.
...
&Chapter1;
&Chapter2;
```

XML is “just” giving you the syntax...

XML does not define any presentation related information...

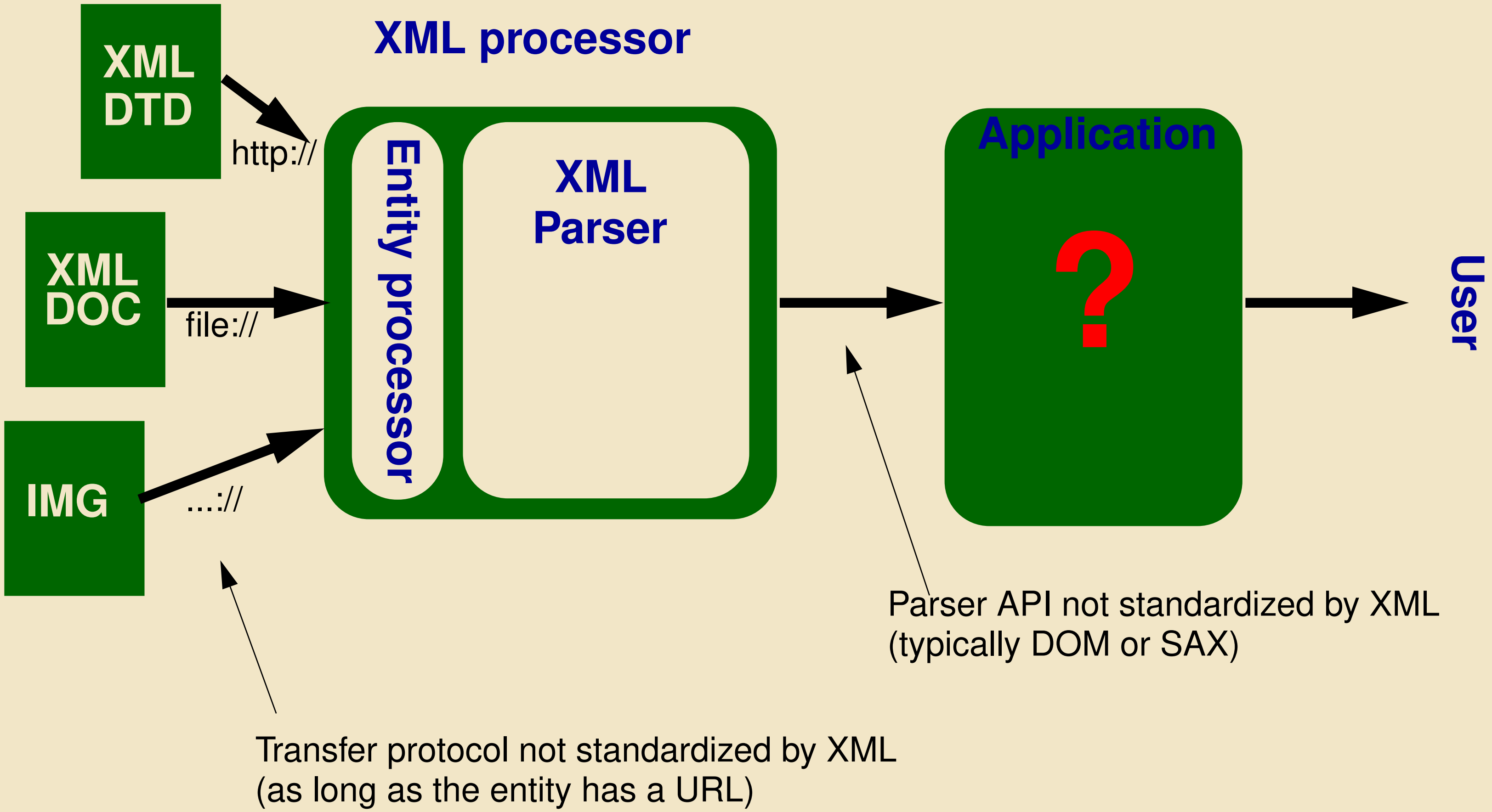
... the XML processor does not know how to present your document!

By the way: it also knows nothing about

- images
- hyperlinking
- meta data
- scripts
- ... or any other semantics attached to your markup

XML is not a markup language

XML processing pipeline revisited



How does the application know what to do?

Presentation

- Built-in presentation semantics
- Style sheets (CSS etc)

Linking

- Built-in link semantics
- Xlink - also used for including images (work in progress)

Meta data

- Resource Description Framework

Transformation

- Transformation sheets (XSLT, work in progress)

Miscellaneous

- Built-in semantics
- Scripts (client or server side)
- ...

Client versus server side processing

Client side processing of XML

- client processes the XML document directly
- end-user can control the presentation
- client needs an XML processor, style sheet engine etc
- all semantic markup available to the end-user
- typical usage: intranet, extranet

Server side processing

- server processes XML and serves formatted data to the client (e.g. HTML)
- content-provider controls the presentation
- client does not need an XML-enabled browser
- semantic markup not available
- typical usage: internet

The XML family and other relatives

	W3C/HTML		
Markup	HTML		
Style	CSS		
Linking	<a href=		
Pointing	<a name=		
Doc. model	DOM		

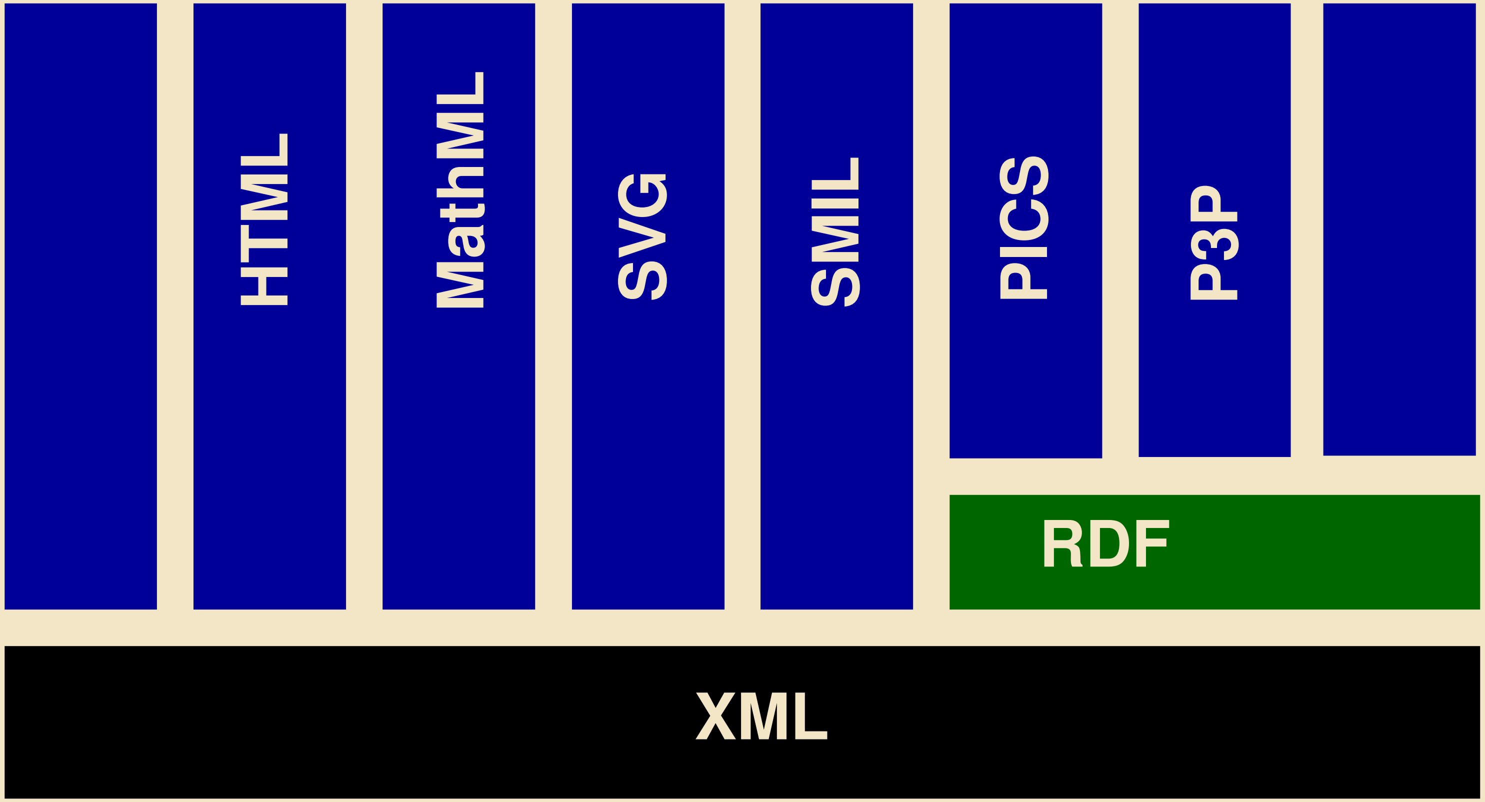
The XML family and other relatives

	W3C/HTML		ISO/SGML
Markup	HTML		SGML
Style	CSS		DSSSL
Linking	<a href=		HyTime/TEI
Pointing	<a name=		HyTime/TEI
Doc. model	DOM		Grove

The XML family and other relatives

	W3C/HTML	W3C/XML	ISO/SGML
Markup	HTML	XML	SGML
Style	CSS	CSS *XSLT*	DSSSL
Linking	<a href=	*XLink*	HyTime/TEI
Pointing	<a name=	*XPEL* *XPointer*	HyTime/TEI
Doc. model	DOM	DOM *SAX*	Grove

Web languages and applications



Future directions

XML is expected to remain stable

- no immediate need for XML 2.0

Many upcoming standards specifying specific semantics

- Linking, transformations, pointing, meta-data, graphics, time, ...

Many new document types using XML syntax

- MathML, SMIL, XHTML, SVG, ...

Many XML applications will need style sheets (CSS2 and beyond...)

- see Bert Bos's talk

HTML will become XML-compliant

- see Steven Pemberton's talk

There is still room for SGML

- see Wim van Halteren's talk

Web sites will fully exploit XML's potential for semantic markup

- see Frank van Harmelen's talk

More pointers to XML

XML 1.0 Recommendation:

- www.w3.org/TR/REC-xml

XML working group:

- www.w3.org/XML/

XML FAQ:

- www.ucc.ie/xml/

XML.com:

- www.xml.com
- www.xml.com/axml/testaxml.htm (annotated XML specification)