What can we do with XML?

6 real and imagined

use cases

SGML/XML Finland ‘98

XML support in browsers

- Can be viewed (like HTML)
- Can be manipulated by applets (like dynamic HTML) with the DOM
- The DOM will be reachable from applets, JavaScript and VBScript

XML server-side

- Many tools exist already to
  - parse XML
  - convert to or from XML
  - generate XML
  - build XML tools
  - store XML in databases
  - import from XML into databases
  - ...

What is new with XML?

- From an SGML perspective:
  - simplicity (AElfred: 23kB, SP: 1.1 MB)
  - wide tool availability
  - widespread understanding and adoption
  - web support
  - many related standards
    - important: validation and parsing are now separate!
  - tool integration

What is new with XML?

- From a web perspective:
  - reliability
  - structured information on the client side
  - more powerful means of structuring information on the server side
  - a standardized basis for exchange formats
  - more advanced linking and location

Using SGML in your tools

- C/C++: Some APIs, but requires large modules
- OmniMark/Balise: not full-featured languages
- Python/Perl/tcl/Lisp: Must read pre-parsed ESIS
- Java: JNI interface to SP
- Eiffel, Sather, Beta, Smalltalk, Delphi…: No products known to me

Using XML in your tools

- C/C++: Several *small* modules
- Java: Lots of parsers, two standard APIs
- Perl: Parser as module, two APIs
- Python: Module and pure parsers, two standard APIs
- tcl: Module and pure parsers, one standard API
- Delphi: Parser component available
- Others: Nothing yet (except Ruby)

So,
what can we do with
all this?

XML Software Autoupdate

Keeping software indexes

up to date

The problem

- Too many products in software indexes for maintainers to check them all for updates
- Too many indexes for developers to keep track of their registrations
- Maintainers need a way to automatically discover new releases and address changes
- Developers need a way of easily telling all interested maintainers about new releases

The solution

- Product home pages already hold all the version and address information needed
- Encoding this information in XML makes the information accessible to software
- XSA lets developers concisely describe all their products in a single file

An XSA document

```
<xsa>
  <vendor>
    <name>James Clark</name>
    <email>jjc@jclark.com</email>
    <url>http://www.jclark.com/</url>
  </vendor>

  <product id=”SP">
    <name>SP</name>
    <version>1.3</version>
    <last-release>19980312</last-release>
    <info-url>http://www.jclark.com/sp.html</info-url>
    <changes>These are the changes since version 1.2:
    ...
```

Using XSA: Developers

- Write XSA document
- Publish it on the web and make the URL known
- Update it whenever something changes

Using XSA: Maintainers

- The XSA document can be monitored by clients that poll the document
- This allows clients to detect:
  - new versions
  - new products
  - address changes (email, home pages etc)

Why is this good?

- For developers:
  - a single file stores all the information needed
  - easy updates: just edit the XSA document
- For list maintainers:
  - a list of URLs and the software is all that is needed
  - checking can be 100% automated

XSA software

- A CGI wizard for making XSA documents
- A validation kit for validating them
- A client kit for monitoring them
  - includes an API for developing custom clients

The future

- XSA will extract its data directly from product home pages (in XML)
- This will avoid duplicating information, but will require namespaces or architectural forms
- Software descriptions may be linked in, to provide searching

What’s new with XSA?

- The ability for software to automatically extract information from web documents and use it
- This is a very important application area that is likely to become much more important with time

Similar systems

- Automatic repeats of news headlines from news sites
- Search sites for a collection of related shopping sites and other catalog sites
- The ICE standard attempts to enable these kinds of things

Why not SGML?

- None of SGMLs extra features are needed
- SGML parsers are not truly platform-independent, making toolkit deployment difficult
- SGML will not be supported on the web, ruling out the more advanced solution

Tax forms in XML

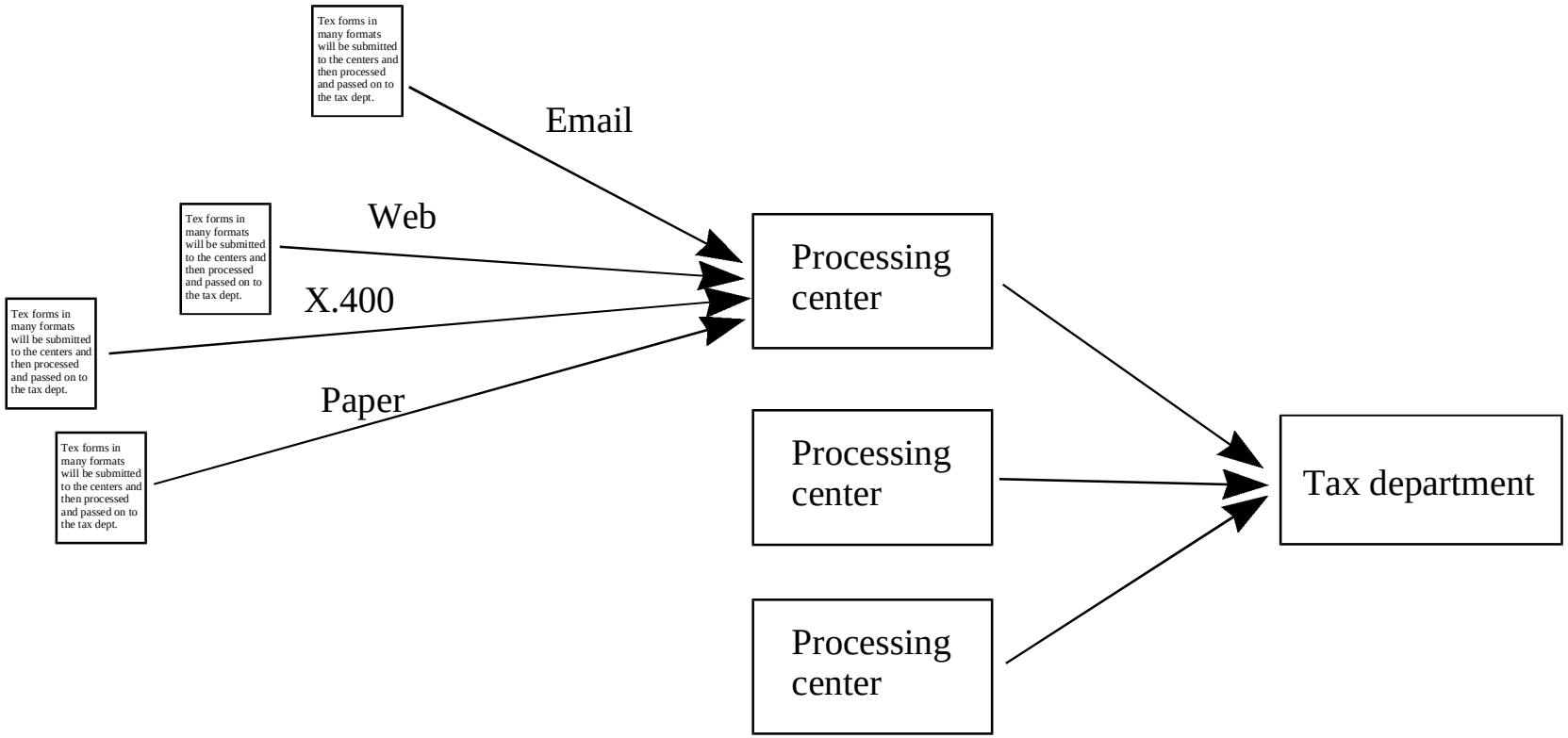
Control information

in XML

Background

- Every year all adult Norwegians submit a tax form with financial information used to calculate the amount of tax to be paid
- Processing such amounts of paper requires enormous resources
- A project to allow these forms to be submitted electronically has been started

Processing model



One way of using XML

- The most obvious way to use XML in this project: store the form data in XML
- This allows using a single submission format that is easy to parse, validate and generate
- The same format can then be used for delivery to the tax department

Another way

- Tax laws change almost annually, and the forms must follow them
- This requires modifications to:
  - the validation software
  - the form input software for all platforms and submission methods
- Modifying source code is expensive...

Solution

- Store the form fields and their relationships as XML:
  - field 1 is ‘Name’, field 2 is...
  - field 15 is the sum of fields 11-14
  - the value of field 17 should be 0 or larger than 15’000,-
  - if field 18 is ‘yes’, fields 19-22 must be filled in
  - ...

Improvements

- This allows:
  - validation software to be generalized, so that it uses the XML document to validate form contents
  - generalization of input software, to display forms as described by the document and also to do validation before submission

What’s new here?

- The idea of using XML documents to define custom validation requirements
- The idea of letting the same XML document control GUI generation
- In other words: to let markup customize or control applications

Why not SGML?

- Again: extra SGML features not needed
- SGML parsers are large, and not available in all languages
- Using full SGML in an applet is not possible
- In general: SGML software is hard to integrate

Recipe Markup Language

An advanced web site

Recipe Markup Language

- Fictional markup language for recipes with:
  - metadata (complexity, time to prepare, country of origin, kind of dish …)
  - ingredients (amounts and alternatives)
  - steps to prepare (with alternatives for ingredients and tools)

The website

- Lets users find and view recipes
- Recipes can be searched for, using metadata:
  - give me an Italian soup that’s easy to make in no more than 30 minutes...
- With nutritional and price databases:
  - ...and costs less than 50 FIM and has less than 500 calories

Personalized service

- With registered users:
  - menu generation based on preferences (price, complexity, tastes, nutritional value…)
  - shopping list generation based on the menus and preferences (shopping frequency, refrigerator size)

Customized display

- A Java applet could display the recipe
- The user could fill in
  - ingredient alternatives
  - equipment alternatives
  - skill level
- The applet could update nutritional values, preparation times and recipe text accordingly

Cooking assistance

- During cooking the applet could
  - leave out irrelevant information
  - show only the current step
  - warn about things that need to be done, such as
    - pre-heating
    - steps that need to be started soon
    - things that have been heated long enough

Science fiction

- In the future, most kitchen appliances are likely to have CPUs and IP-addresses
- The Java applet in your web browser could then actually monitor your cooking
- You could also search for recipes that could be made with the ingredients in your refrigerator without typing them in

Making money

- Selling access to
  - the personalized service
  - the display applet
- Selling the recipes
  - as a book
  - on CD-ROM
- Selling targeted advertisements

Why is this interesting?

- A general example of the services that become possible with structured information
- Similar things can likely be done with other kinds of structured information
- Not possible with a plain RDBMS solution

Genealogical data in XML

XML as an
exchange format

The situation today

- Genealogical research is a common hobby
- Lots of programs exist to help researchers
- Data can be exchanged using the GEDCOM exchange format
- Web integration with GEDCOM is very poor

GedML

- XML DTD developed by Mike Kay of ICL
- Software exists to convert to and from GEDCOM and CSV files
- Not much more has happened, yet
- Note that some of the things I describe in the following are not possible because of things inherited from GEDCOM

Supporting GedML

- All programs that can accept and export GedML can exchange data
- Developing this is easy because:
  - XML parsers already exist
  - XML is easy to generate
  - The GedML DTD provides a formal and useful specification of the format

Using GedML

- GedML can be published on the web and
  - read with a style sheet
  - navigated with an applet
  - branches can be imported into your own data
  - searched by GedML search engines

Going further

- Packages could be developed to turn a GedML document into a genealogical part of a site with
  - site map
  - search engine
  - family trees
  - ...

Even further

- Online GedML files could use XLink to link into other GedML files
- XPointers could be used to pinpoint individuals and families within other files
- This could be used by different researchers to cooperate on different family branches
- Browsing software could follow the links

Searching

- Standard search engines could be used to search for GedML files containing certain strings
- Search engines could specialize in genealogical searches and do web-wide searches

XML as an exchange format

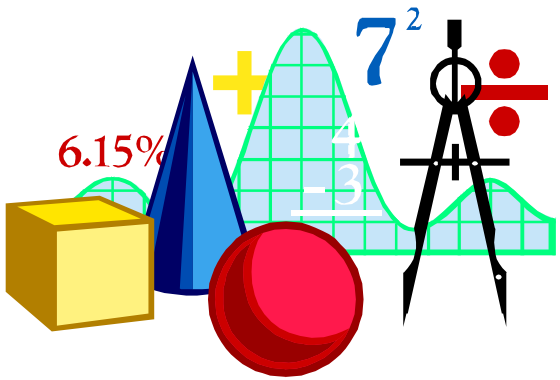
- Better than SGML because
  - it’s simpler
  - there are more tools
- Better than binary formats because
  - it’s simpler
  - it’s human-readable
  - can be modified in a text editor
  - easier to parse

XML as an exchange format

- XML provides an entire infrastructure for:
  - parsing
  - validation
  - syntax description
  - display
  - object models
  - linking and location
  - web deployment

MathML

Math on the web



MathML

- Browsers had no capabilities for showing mathematical formulae except as images
- To solve this the W3C has developed an XML DTD for representing formulae: MathML
- MathML can be edited with special editors and manipulated in Java applets

Using MathML

- The most obvious way: displaying math
- Exchanging of formulae between math software packages such as
  - engineering systems
  - equation editors
  - web pages
  - educational software

Drag and drop

- Engineers could drag formulae from technical specs in web pages and into their engineering software
- Students could drag formulae from educational web pages and into software packages to analyze them, modify them and display graphs

Using MathML II

- An API already exists for developing applets that work with MathML in browsers
- This can be used to
  - display MathML
  - edit MathML
  - use MathML in calculations or to display graphs

Linking

- Applets can be use XPointers to locate formulae to display in some way
- Scientific papers and educational texts can link directly to
  - the proofs or derivations of the formulae they use
  - alternate forms of the formulae
  - ...

Current MathML support

- IBM Techexplorer, tech document viewer
- MathType, equation editor used in MS Office, WordPerfect, ClarisWorks and Nisus Writer (in version 4.0)
- Maple, engineering computing system
- Mathematica, typesetting system
- Publicon, publishing system (not yet)

What’s interesting about this?

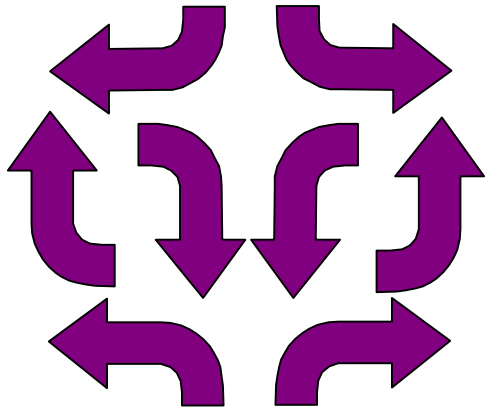
- An example of a general format that can be used for several different things:
  - access to structured data in browsers
  - exchange
  - display/publishing
- More formats of this kind will very likely be developed in the future

Why not SGML?

- MathML actually started as an SGML project, but since XML will be supported by browsers, SGML was dropped
- Again: simplicity and tool availability

Site maps

New ways of browsing



Browsing today

- Most commercial sites carry lots of navigational links on every web page
- Despite this they are often hard to navigate
- Searches often return results like:
  - slides 23, 35 and 54 of a presentation
  - sections 2.1, 3.4, 4.1 of a paper
  - ...

Resource Description Framework

- A framework for making metadata languages with XML syntax
- Not a finished W3C recommendation, yet
- The framework can be adapted to many different purposes, such as
  - describing software
  - site maps
  - ...

Site maps in RDF

- This could be a single XML document that described
  - each page on a site
  - the relationships between the pages
  - the site as a whole
  - the relationships to other sites

Searching with site maps

- Search engines could use them to direct searches
  - display the context for a hit (what kind of document, links to other pages on the site)
  - group hits that are logically within a single resource
  - inform about related sites

Navigation with site maps

- With a site map the browser interface could be much improved
- Instead of the simple backwards/forwards model, the browser window could be split in two:
  - site map with ‘you are here’
  - the page itself

Consequences

- Web navigation could be much simpler
- Web site maintainance could be much easier
- Web page layout could become much more readable and user-friendly
- Browsers as we know them might change completely

Conclusion

- This is another example of the new possibilities that open up with structured information on the web
- Why not SGML: not supported on the web and not needed

Conclusion: Prelude

How can XML be both
simpler than and more powerful than
SGML?

“Worse is better”

- In 1991 Richard P. Gabriel wrote an analysis of the current situation for the programming language Lisp
- In it he described two design philosophies:
  - “The right thing”
  - “Worse is better”

Two design philosophies

- “The right thing”:
  - correctness and completeness over simplicity
  - simplicity of interface over implementation
- “Worse is better”:
  - simplicity of implementation above all, especially completeness

Examples

The right thing

- Common Lisp/Scheme
- X.25 + X.400
- Xanadu
- SGML
- HyTime
- DSSSL

Worse is better

- C/C++
- TCP/IP + SMTP
- WWW
- XML
- XLink
- CSS2/XSL

Consequences

- More people understand XML
- More XML software
- XML software requires less resources
- More XML standards
- More XML publicity
- XML will be extended with time

Conclusion

What can we do with XML?

What does it do better than SGML?

What is XML?

- A standard
- A way of thinking
- An infrastructure of existing
  - tools
  - professionals
  - techniques
  - supporting standards

Impact on SGML

- Wider adoption
- More publicity
- New ways of using markup
- New schema standards
- New usage areas

Impact on the web

- Simplifies data exchange
- Enables entirely new kinds of applications
- “Gives Java something to do”
- A cornerstone of the new web architecture
- Simplifies site maintainance

New uses for XML

- XML as configuration and data format for applications
- XML as simple exchange format
- XML as basis for automatic information extraction from web pages
- XML as structured data in browsers, enabling more advanced web services

Is SGML dying?

- There are more SGML documents than HTML documents…
- Organizations like Boeing, Microsoft, Sybase, Mobil, US Armed Forces etc have built their information systems on SGML
- SGML can do things XML cannot
- In a word: no

How to use SGML with XML

- Publishing XML from SGML is easier than publishing HTML from SGML
- James Clark's SX does this automatically with no configuration at all
- HyTime, TEI Extended Pointers etc will still need to be converted
- Conversion can do this

The roles of XML and SGML

XML

- Web publishing
- Exchange
- Simple in-house use
- Configuration files
- Data serialization
- Data formats

SGML

- Complex in-house use
- Long-term storage
- Hand-edited information

The next generation will do things with SGML that we can’t even imagine yet - it is that versatile.

William W. Davis,
chairman ANSI SGML
Task Group, April ‘95