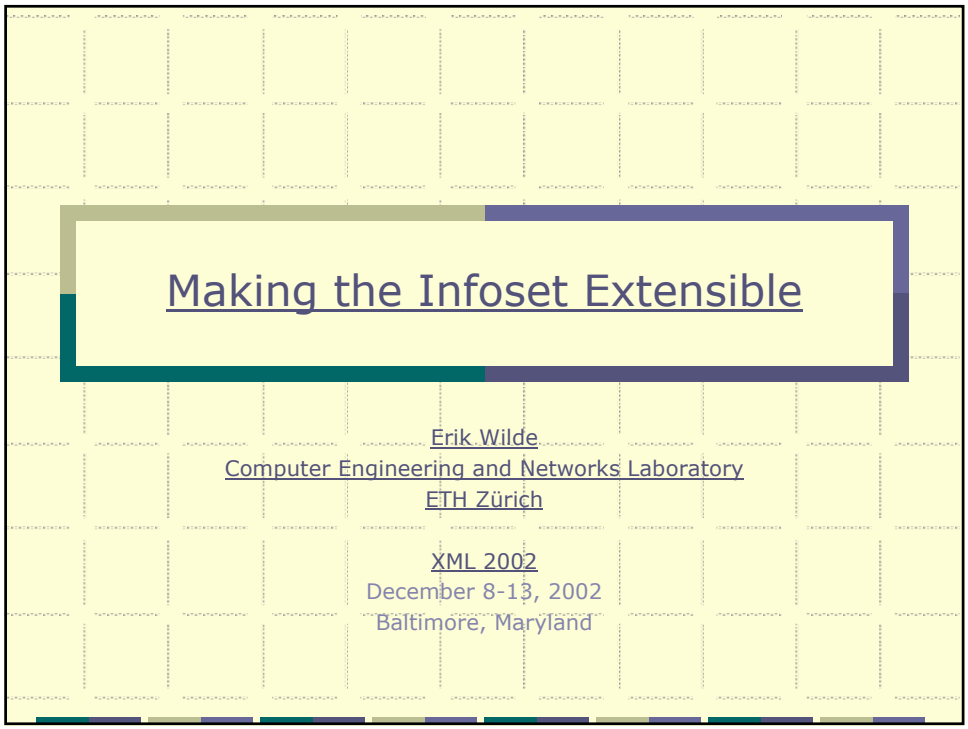

Making the Infoset Extensible
Erik Wilde
Computer Engineering and Networks Laboratory
ETH Zürich
XML 2002
December 8-13, 2002
Baltimore, Maryland


Abstract
The XML Infoset defines the data model of XML, and it is used by a number of other specifications, such as XML Schema, XPath, DOM, and SAX. Currently, the Infoset defines a fixed number of Information Items and their Properties, and the only widely accepted extension of the Infoset are the Post Schema Validation Infoset (PSVI) contributions of XML Schema. XML Schema demonstrates that extending the Infoset can be very useful, and the PSVI contributions of XML Schema are being used by XPath 2.0 to access type information in a document's Infoset.
In this paper, we present an approach to making the Infoset generically extensible by using the well-known Namespace mechanism. Using Namespaces, it is possible to define sets of additional Information Items and Properties which are extending the core Infoset (or other Infoset extensions, defining a possibly multi-level hierarchy of Infoset extensions). Basically, a Namespace for an Infoset extension contains a number of Information Items, which may have any number of Properties. It is also possible to define an Infoset extension containing only Properties, extending the Information Items of other Infosets.
Further elaborating on this method, many of the XML technologies currently using the Infoset could be extended to support the Infoset extensions by importing Infoset extension using the extension's Namespace name. To illustrate these concepts, we give an example by defining the XML Linking Language (XLink), the XML vocabulary for hyperlinking information, in terms of Infoset extensions. We show how the proposed ways of supporting Infoset extensions in XML technologies such as XPath, DOM, and CSS could pave the path to a better support (and hopefully faster adoption) of XLink than we see today. XLink serves as one example, but the proposed extensions and techniques are not limited to this particular technology.
The content of this paper is work in progress, contributing to the ongoing debate on how to deal with different XML vocabularies and their usage in other XML technologies. We believe that making the Infoset extensible would provide a robust and flexible way of making the data model of XML-based data more versatile, and creating an accepted way of making the data available through standard interfaces such as DOM and XPath.
XML 2002
Making the Infoset Extensible (©2002 Erik Wilde)
2

Outline

1. The Infoset does not change
2. Motivation
3. Current State of the XML Infoset
4. Examples
5. Changes to the Infoset spec
6. Further Work & Conclusions
7. Q&A

The Infoset does not change!

- XML Infoset is the heart of many specs
  - APIs: DOM3, SAX2
  - addressing into XML: XPath
    - used by XSLT, XML Schema, XQuery, XPointer
  - XML normalization: Canonical XML
- XML Infoset is a simple spec
  - 11 properties, each with a number of properties
- *Extensible XML Information Set (EXIS)*
  - defines the exact same properties…
  - but defines them in a more formal way
    - using an XML-based "Infoset schema" language
  - also defines an extension mechanism

Motivation I (XML Schema)

- XML Schema works with the Infoset
  - formally, schema validation augments the Infoset
  - *Post Schema Validation Infoset (PSVI) Contributions*
    - http://www.w3.org/TR/xmlschema-1/#PSVI_contributions
  - i.e., validation is using existing XML specifications
- how to access validation results
  - there is no standard way
    - DOM only provides access to the "core Infoset"
  - Infoset additions are possible, but unsupported
    - DOM, SAX, XPath do not support Infoset additions
  - parsers define their own APIs to PSVI information
    - Apache's PSVI Interface for DOM and SAX
- XML Schema PSVI information is hard to use

Motivation II (XLink)

- XML and hyperlinking
  - XLink/XPointer development by W3C
  - XLink for embedding linking information in XML
- XLink adoption is slow
  - lack of support by software such as browsers
  - lack of support for software authors
    - how to program with XLink? how to style XLinks?
  - XLink integration in other languages is a problem
    - XHTML currently uses HLink rather than XLink
- XML hyperlinks need a better foundation
  - a well-defined data model endorsed by W3C
  - syntaxes (XLink/HLink) and interfaces (API/UI) as necessary

An Open XML Data Model

- XLinks extends XML's data model
  - additional information about resource fragments
  - it only uses XML syntax for link representation
- link information may come from any source
  - from XLinks within the document
  - from XLinks retrieved from an external linkbase
  - from non-XLink information from any data source
- currently links are not reflected in the Infoset
  - programming with links is cumbersome
    1. find all the relevant information (via the Namespace)
    2. check for integrity as defined by XLink's constraints
    3. process the link information

XML 2002 Making the Infoset Extensible (©2002 Erik Wilde) 7

XML Infoset Essentials

- only Namespace-compliant XML allowed!
  - some XML documents do not have an Infoset
- what is in the Infoset?
  - elements (child sequences) and attributes (sets)
  - Namespace declarations and prefixes
  - comments and processing instructions
  - character data
  - some type information (attribute types)
- and what is not in the Infoset?
  - whitespace within tags
  - the order of attributes within element tags
  - most information from the DTD

XML 2002 Making the Infoset Extensible (©2002 Erik Wilde) 8

XML Information Set

- several XML applications need a data model
  - APIs (DOM, SAX)
  - XML programming languages (XSLT)
  - XML query languages (XQuery, XQL, …)
  - XML formatting (CSS, XSL-FO)
  - XML fragment identifiers (XPointer)
- XML 1.0 does not define a data model
  - implicitly defined, but open to interpretation
    - does attribute order matter?
    - does whitespace in tags matter?
- XML Information Set (XML Infoset)
  - defines a set of *information items* with *properties*
  - an XML document is a set of such items

Infoset vs. Document

- the Infoset is a rather abstract construct
  - it does not have a particular representation
  - it does not have a particular interface
  - it often is produced by a parser
    - but it may also be synthesized
- the Infoset is more than a parse tree
  - it omits some information
    - syntax details deemed irrelevant
  - it adds some information (schema-derived)
    - type information about attributes
    - default attributes from a schema
    - references for `ID/IDREF(S)` attributes
- Infoset = Abstraction + Semantics

Additional Information

- Infoset represents XML markup data
  - the "relevant" parts of an XML document
  - additional semantics may be present
- additional information may be required
  1. obtained by interpreting the markup
     - e.g., recognizing XLink attributes
  2. obtained by adding external information
     - e.g., links retrieved from linkbases
     - e.g., type information from an XML Schema
- may be important for processing
  - should be preserved through processing steps
  - should be represented generically

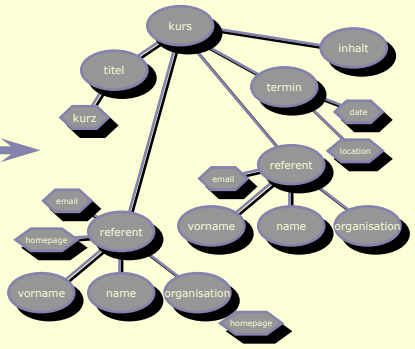
Several XML data models

- W3C develops a number of data models
  - Infoset intended as a foundation for other specs
  - DOM/SAX for APIs
    - DOM3 will be aligned with the Infoset
  - XPath 1.0 for XSLT 1.0 and XPointer
    - a stripped-down version of the Infoset
  - PSVI including XML Schema validation results
    - extending the Infoset with PSVI contributions
  - XPath 2.0 for XSLT 2.0 and XQuery
    - XPath 1.0 plus PSVI information for typing
  - XQuery extending the XPath 2.0 data model
- it's hard to work with all of them
  - but often necessary when combining components

Where does it come from?

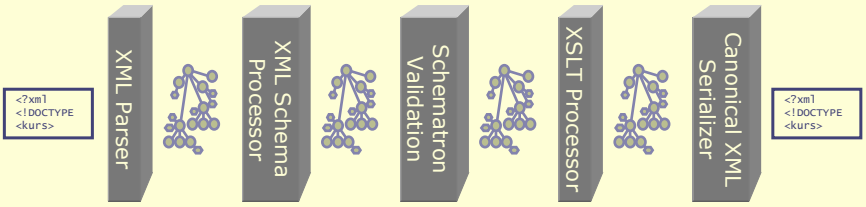
```
<?xml version="1.0" ?>
<!DOCTYPE kurs SYSTEM "kurs.dtd">

<kurs>
<titel kurz="XML">XML -
Grundlagen und Umfeld</titel>

<referent email="xml@dret.net"

homepage="http://dret.net/">
  <vorname>Erik</vorname>
  <name>Wilde</name>
  <organisation
homepage="...">ETH
  Zürich</organisation>
</referent>

<referent> ... </referent>
<inhalt> ... </inhalt>

</kurs>
```

XML Parser



Where does it go?

- XML processing as a DOM Pipeline
  - many analogies to the Unix pipe metaphor
  - small, specialized, modular components
  - combined to achieve complex processing results




```
parse x.xml | xsv | schematron | xslt -s test.xsl | serialize -can > y.xml
```

What's wrong with this picture?

- nothing! a very useful way of XML processing
  - Sourceforge's XPipe project
  - Sun's "XML Pipeline Definition Language" W3C note
- but: a DOM pipeline restricts the data model
  - only XML markup structures can be represented
  - it is impossible to pipe "augmented Infosets"
- what we want is an Infoset pipeline
  - supporting augmented Infosets
    - e.g., including links retrieved from a linkbase
  - supporting pre-/postconditions for Infoset extensions

```
parse x.xml | linkify | lbap lb.xlb | serialize > y.xml
```

XML 2002 Making the Infoset Extensible (©2002 Erik Wilde) 15

Analyzing the Pipeline

```
parse x.xml | linkify | lbap lb.xlb | serialize > y.xml
```

- parsing needs only basic XML processing
  - maybe validation using a DTD
- linkifying requires XLink knowledge
  - turns XLink elements into XLink Infoset contributions
- Linkbase Access Protocol (LBAP)
  - accepts an XLink Infoset
  - retrieves external links (optionally accepts linkbase)
    - XLink markup may also point to linkbases
  - produces an XLink Infoset containing external links
- serialize produces XML markup
  - transforms XLink Infoset items to XLink markup

XML 2002 Making the Infoset Extensible (©2002 Erik Wilde) 16

EXIS

- Extensible XML Information Set (EXIS)
  - a reformulation of the XML Infoset
  - does not change the XML Infoset
- EXIS explicitly supports Infoset extension
  - using a formal notation for Infoset extensions
    - making it possible to process the information
  - defining a schema language for Infoset extensions
    - basically, a notation for items and/or properties
  - defining rules for Infoset extensions
    - they are identified by a namespace name
    - they must be based on other Infoset extensions
      - in the simplest case, on the "core Infoset"
    - there can be no cyclic dependencies

Applications Supporting EXIS

- applications supporting EXIS must register
  - the namespaces of the supported extensions
  - framework ensures proper EXIS component pipelines
- EXIS transformers for reducing/enriching
  - reducing the Infoset by stripping extensions
    - e.g., stripping PSVI contributions from the Infoset
    - e.g., transforming XLink Infosets to XLink elements
    - e.g., transforming XLink Infosets to XHTML
  - enriching the Infoset by creating extensions
    - by interpreting markup (e.g., XLink)
    - by adding external information (e.g., PSVI)
- EXIS applications as XML pipeline components

Infoset Extension: Example I

- XML Schema extends the Infoset
  - Infoset augmentation (adding items and properties)
  - validation results are available through the Infoset
- PSVI is hardwired into other W3C specs
  - the XPath 2.0 data model uses PSVI contributions
  - currently there is no API for PSVI
    - no active work on PSVI DOM module
    - validation is easy, using the results is not!
- PSVI/EXIS could be accessed using standards
  - generic mechanisms in DOM/SAX/XPath
  - specific mechnanisms such as a DOM EXIS module

Infoset Extension: Example II

- XLink is defined in terms of XLink syntax
  - XML philosophy: syntax explicit, model implicit
    - an excellent topic to argue endlessly
  - at some point, the model is necessary
    - are simple/extended links different links?
    - or just different syntaxes for a link? how do I choose?
    - is the distinction simple/extended visible in the model?
- XLink 1.1 should define a linking data model
  - in terms of Infoset augmentation
  - e.g., as an EXIS extension
- XLink 1.1 should also define a syntax
  - as something separate from the data model
  - others may dismiss the syntax, but keep the model

Generic Infoset Extensions

- defining Infoset extensions is not very hard
  - define a Namespace URI for the extension
  - define possible dependencies with other extensions
  - define the Infoset extensions
    - additional *properties* of existing *items*
    - additional *items* defines by the extension
- extensibility should be supported everywhere
  - define a DOM module for generic Infoset extensions
    - if better support is required, define a specific module
  - define XPath mechanisms for using extensions
    - for example, define "extension axes" for XPaths
  - define CSS selectors for extensions
    - CSS currently is based on XML rather than the Infoset

The 80/20 Split

- many people don't like the idea of extensibility
  - things get more complicated
  - software must be more robust
  - and: many people don't like Namespaces
- is it worth the effort?
  - depends on the point of view…
    - if you are happy with plain XML, it is not
    - if you see XML as a foundation for more advanced data models supporting additional semantics, it is
- political questions are important
  - W3C has very many people from a lot of companies with very different goals and backgrounds

Future Work

- closer look at some questions of support
  - is a generic API (DOM/SAX/JDOM) doable/reasonable
  - is there a clean way of integration into XPath?
- schema for Infoset extensions
  - which schema language to use?
    - (XML Schema | RELAX NG) & Schematron
  - reformulate Infoset as "core Infoset" EXIS module
- how to efficiently pipeline EXIS data
  - naïve but easy: EXIS XML syntax (extremely big)
  - Infoset XML Schema as a starting point
  - DOM/Infoset pipelines should use other mechanisms
    - shared memory or at least an efficient PDOM+EXIS

Conclusions

- EXIS as a way to look beyond markup
  - sometimes unnecessary
  - sometimes very useful
  - can always be thought of as optional
- attempt to tame the zoo of XML data models
  - at least they will be easier to read
  - dependencies will be made more explicit
  - processing can use generic mechanisms (DOM EXIS)
- XML standardization is complex
  - hard to talk to the right people
  - hard to convince all the necessary people
  - hard to coordinate very diverse efforts and groups