Quo vadis XML?
“Do we need new syntax, data model or both or is XML just fine after 15 years?”

XMLPrague CFP
Celebrating XML's birthday

XML is now 15 years old...

You are here!

1998  2013
Celebrating XML's birthday

During this time...

- ...XSLT and XPath were invented and extended and modified up to version 3.0
- ...XQuery was developed
- ...XML Schema 1.0 and 1.1 were standardized
- ...RELAX NG was built on the shoulder of RELAX and TREX
- Schematron was standardized

You are here!
Celebrating XML's birthday

But XML itself...

- ...maintained more or less stable – we have five editions of XML 1.0 and two editions of XML 1.1 (which was not a huge success)

You are still here!
Why is XML so stable? So mature? Maybe the answer lies in its history...
XML's heritage

- First considerations about generic markup have been made in the 60s such as GenCode, Text Description Language (TDL, later GML)
- In the 1970s GenCode and GML combined forces
The 80s: SGML

- In the 80s, SGML was standardized as ISO standard
- SGML as a meta language laid the foundation for a whole family of markup languages such as
  - TEI
  - Docbook
  - HTML up to 4.01
The 90s: Lots of Pulp – and HTML and XML

- HTML was developed in the early 90s based on SGML
- XML was created as an SGML subset in 1997 and standardized as W3C Recommendation in 1998 to conquer the web
- As an SGML subset it inherited most basic components
Basic concepts of a markup language

In 1999 Sperberg-McQueen and Huitfeldt named three basic concepts of SGML

- Linearization
- Data model
- Constraint language

In 2004 the authors coined the tripod metaphor
Basic concepts of a markup language

Witt added 2004 the distinction between

• a level (the concept that drives certain markup) and

• the layer (the serialization of the markup)
  → Think of microformats or TEI feature structures

• This distinction allows for generic markup – and a quatropod metaphor
Basic concepts of a markup language

But we can go even further:

• We can split up linearisation into
  – Notation (inline vs standoff) and
  – Syntax (this is especially interesting when dealing with non-XML-based markup language)
In terms of XML

- XML's data model is tree-like → let's stick with the term hierarchy
- XML's syntax consists of elements, attributes and tags (amongst others) and uses special characters to differentiate between markup and content
- XML's notation can be inline or standoff
- Various constraint languages are available
- Level/layer abstraction is grammar-driven
XML is 15 – that is, it's a teenager

Teenage problems:

• Teenagers test boundaries – e.g. boundaries of the data model, such as overlapping markup

• Teenagers do not behave as expected – take the different requirements regarding deterministic content models in DTD, XSD and RNG

• Teenagers challenge traditional values – such as draconian error handling in parsing (see HTML5)

• Teenagers try to mimic others – such as JSON's data model (wait for Eric van der Vlist's talk)
How to cope with teenage problems

Try to educate them

• Find a way to cope with its limitations
  → either try to simplify or extend XML's components without losing compatibility

Start over again

• Tell the teen to get a life – and invent alternative approaches which address at least one of XML's limitations
Approaches that are compatible to teen XML

- MicroXML
- XStandoff
- xLMNL
- ...

Approaches that are incompatible to teen XML

- LMNL
- FtanML
- ...

...
MicroXML

- We won't go into details here since we've already heard Uche Ogbuji's talk
- MicroXML addresses the following of XML's components:
  - Data model
  - Syntax
  - Parsing
XStandoff

• A hybrid meta markup language that was developed since 2008 to allow multiple (and possible overlapping) annotations – especially linguistically annotated data

• Support for discontinuous elements, multiple parenthood and differentiation between dominance and containment

• XStandoff addresses the following of XML's components:
  - Data model (graph- or GODDAG-like)
  - Level/layer distinction
XStandoff – A simple example

<?xml version="1.0" encoding="UTF-8"?>
<morphemes xmlns="http://www.xstandoff.net/morphemes">
  <morpheme>The</morpheme>
  <morpheme>sun</morpheme>
  <morpheme>shine</morpheme>
  <morpheme>s</morpheme>
  <morpheme>bright</morpheme>
  <morpheme>er</morpheme>.
</morphemes>

<?xml version="1.0" encoding="UTF-8"?>
<syllables xmlns="http://www.xstandoff.net/syllables">
  <syllable>The</syllable>
  <syllable>sun</syllable>
  <syllable>shines</syllable>
  <syllable>brigh</syllable>
  <syllable>ter</syllable>.
</syllables>
XStandoff – A simple example

<?xml version="1.0" encoding="UTF-8"?>
<morphemes xmlns="http://www.xstandoff.net/morphemes">
  <morpheme>The</morpheme>
  <morpheme>sun</morpheme>
  <morpheme>shine</morpheme>
  <morpheme>s</morpheme>
  <morpheme>bright</morpheme>
  <morpheme>er</morpheme>.
</morphemes>

<?xml version="1.0" encoding="UTF-8"?>
<syllables xmlns="http://www.xstandoff.net/syllables">
  <syllable>The</syllable>
  <syllable>sun</syllable>
  <syllable>shines</syllable>
  <syllable>brigh</syllable>
  <syllable>ter</syllable>.
</syllables>
The sun shines brighter.
The sun shines brighter.
The sun shines brighter.
The sun shines brighter.
The sun shines brighter.
The sun shines brighter.
The sun shines brighter.
XStandoff and (x)LMNL

• XStandoff extends the limits of XML's data model but otherwise remains in the XML world

• Similar to that is LMNL's (Layered Markup and Annotation Language) XML serialisation by Tennision introduced

• LMNL addresses the following XML components:
  – Syntax (when using the sawtooth syntax)
  – Data model (Ranges instead of hierarchies)
  – Validation (depending on the syntax used)
LMNL

LMNL differs from XML

- LMNL uses (possibly overlapping) ranges (sequences of characters) instead of hierarchies
- Ranges can be annotated
- LMNL supports structured annotations (annotations can be miniature documents – think of structured attributes in plain XML)
- In contrast to XML attributes LMNL annotations are ordered and more than one annotation of the same name and range can be given
And he said unto them, "What counsel give ye that we may answer this people, who have spoken to me, saying, "Make the yoke which thy father did put upon us lighter?"

And the young men that were grown up with him spake unto him, saying, "Thus shalt thou speak unto this people that spake unto thee, saying, "Thy father made our yoke heavy, but make thou it lighter unto us;" thus shalt thou say unto them, "My little finger shall be thicker than my father's loins."

And now whereas my father did lade you with a heavy yoke, I will add to your yoke: my father hath chastised you with whips, but I will chastise you with scorpions.

So Jeroboam and all the people came to Rehoboam the third day, as the king had appointed, saying, "Come to me again the third day."
And he said unto them, What counsel give ye that we may answer this people, who have spoken to me, saying,
And he said unto them, What counsel give ye that we may answer this people, who have spoken to me, saying, 

-- [...] -->

1 Kings
And he said unto them, What counsel give ye that we may answer this people, who have spoken to me, saying,
LMNL

• LMNL shows some interesting extensions to XML's data model – and a possible linearisation

• Since it lacks support for hierarchies it abandons one of XML's data model key features
FtanML

- FtanML was developed during a summer academy by Michael Kay and others.
- It addresses the following of XML's components:
  - Data model
  - Notation format
  - Constraint language
- The data model is similar to JSON's array of key-value pairs but adds elements, attributes and contents.
- In contrast to JSON it supports XML's mixed content.
FtanML

Some examples:

- JSON-like arrays
  \[ 1, 2, \text{‘abc’}, [1, 2] \]

- XML-like FtanML element
  \texttt{<para|Here is some <b|bold> text>\texttt{}}
FtanML

• There is a parser for FtanML and serialization for its object model in JSON and XML but no traversal language or validation language

• Up to now, FtanML is only a proposal – but there will be further development (hint: wait for this year's Balisage)
Quo vadis XML?
Quo vadis XML?

There are different development options:

- If we change nothing we may store any information we want – but it may be cumbersome (think of XStandoff and xLMNL) – but we'll benefit from XML's ecosystem

You are here!

1998  2013  Do nothing
Quo vadis XML?

There are different development options:

- We could go the MicroXML way and both simplify XML and adapt its data model
Quo vadis XML?

There are different development options:

- We can invent another meta language which a certain feature set like LMNL or FtanML – but we may lose the benefits of XML's ecosystem.
I myself cannot answer which way is the best, but I could raise concerns for some points...
What I'd like to keep from current XML

- Hierarchical data model
- Support for mixed content models
- Processable with XML's toolchain (this does not have to mean XML's syntax, since XML's toolchain is capable of processing non-XML input, see Piez' LMNL sawtooth parser written in XSLT 2.0)
- Powerful constraint languages such as XSD 1.1 or RNG
- Powerful transformation and query languages such as XSLT and XQuery
What I'd like to see added

- Support for overlapping hierarchies and discontinuous elements
- Use a modular approach – simple for those that need only a fraction of its power and complex for those that need everything
Thank you for your attention!
Questions? Comments?

maik.stuehrenberg@uni-bielefeld.de
@mstuehrenberg
References

- All cliparts except the 'Pulp fiction' by http://openclipart.org (Public Domain license)
- 'Pulp fiction' SVG conversion taken from http://1.bp.blogspot.com/-mIjch6QirOY/TbbQx4JJQ6I/AAAAAAAABPo/a_ui9B1BnvE/s320/Pulp_Fiction_Wallpaper_by_Oshowom.jpg