*XML Plenary Lightening Talk*

# Making the XML Stack Work with XML 1.1

Noah Mendelsohn
IBM Research
March 3, 2003

# Goal of this talk

- Goal of this talk: alert non XML-core WGs to need to start building an interoperable stack in an XML 1.1 world.
- Non-goals:
  - Promote or deprecate use of XML 1.1 in particular situations or recommendations
  - Debate W3C process: did right reviews happen prior to 1.1 rec?

# What’s New in XML 1.1?

# Noah Buys an Island

**…and designs his own letter used in place of “M”**

English:

Noah Mendelsohn

Islandese:

Noah endelsohn

# What Changes?

| | Allowed in XML 1.0? | Allowed in XML 1.1? |
|---|---|---|
| <XML> | *Yes* | *Yes* |
| **<χμλ>** | *Yes* | *Yes* |
| <X L> | *No* | *Yes* |

# What Changes?

| | Allowed in XML 1.0? | Allowed in XML 1.1? |
|---|---|---|
| <attr="XML"> | *Yes* | *Yes* |
| <attr="χμλ"> | *Yes* | *Yes* |
| <attr="X L"> | *Yes* | *Yes* |
| <attr="&#x1f"> | *No* | *Yes* |

# Some Issues to Consider

# Schema: Declaring Elements

XML Schema

<xsd:element name=“X L”>

**Is this OK?**

**Can you declare elements with the new names?**

# Schema: Declaring Elements

XML Schema

`<xsd:element name="X` `L">`

**Is this OK?**

**Can you declare elements with the new names?**

**No. In an element declaration, the {name} component is specified as "an NCName as defined by [ref to 1999 version of Namespaces in XML]" (from XML Schema Part 1)**

# Schema: String Type

XML Schema

```
<xsd:attribute name=“at”
     type=“xsd:string”>
```

at=“Noah endelsohn”

**Validates.**

# Schema: String Type

XML Schema

```
<xsd:attribute name=“at”
     type=“xsd:string”>
```

~~at=“&#x1f”~~

**Doesn’t validate with xsd:string type.**

***“The ·value space· of string is the set of finite-length sequences of characters (as defined in [XML 1.0 (Second Edition)]) that ·match· the Char production from [XML 1.0 (Second Edition)].”  (XML Schema Part 2)***

# Schema: Name Type

XML Schema

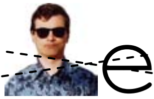
```
<xsd:attribute name=“at”
     type=“xsd:name”>
```

at=“Noah endelsohn”

# Schema: Name Type

XML Schema

```
<xsd:attribute name=“at”
     type=“xsd:name”>
```

~~at=“Noah endelsohn”~~

**Doesn’t validate unless we change the Schema Rec.**

*“The ·lexical space· of Name is the set of all strings which ·match· the Name production of [XML 1.0 (Second Edition)].” (XML Schema Part 2)*

# Query: Functions & Operators

fn:node-name(<abc>)

fn:node-name(<X L>)

# Query: Functions & Operators

fn:node-name(<abc>) **OK**

~~fn:node-name(<X L>)~~ **Doesn't work**

**Doesn't work…the return type of fn:node-name is an xsd:QName which is defined as (you guessed it)…**

***"The ·value space· of QName is the set of tuples {namespace name, local part}, where namespace name is an anyURI and local part is an NCName [I.e. per 1999 version of Namespaces in XML…NRM]" (XML Schema Part 2)***

# But there’s more…

- **How do you know which Infosets allow new names/chars?**
- **application/xml (RFC 3203) needs clarification (informal discussions started)**
- **SOAP is based on RFC 3203…did not explicitly anticipate new content…affects interoperation**
- **Even old-style content may be labeled as <?xml version=“1.1”?>.**
  - Streaming large documents complicates early generation of <?xml version=“1.x”?>
  - Not clear whether query can easily check for new content when serializing
  - XML interop story compromised: already deployed software can’t read even old-style content
- **WSDL?, RDF? Probably others?**
- **I may not have the list right, but there clearly are a lot of interrelated issues to consider.**

# Conclusion

- XML 1.1 seems to affect many Recommendations, RFC’s, etc.
- We need to get busy figuring out the implications
- My opinion: we need a stack that is consistent and that preserves the nearly universal interoperability that has made XML compelling for users.