The XML Enabled Directory (XED)

Implementation Considerations

Steven Legg

eB2Bcom

steven.legg@eb2bcom.com

Daniel Prager

Deakin University

dan@layabout.net

XED Features

- XML encodings for ASN.1 values
  - Robust XML Encoding Rules (RXER)
- XML schema data types referenced from ASN.1
- Extended Component Matching
- User defined directory attribute syntaxes
- XML-ized protocols (e.g. XLDAP)

XML as a Transfer Syntax

- BER, GSER are self-contained at all levels of nesting
- XML namespace declarations are inherited by nested elements
- Entities and notations are declared in the DTD and their scope is the entire document
  - parsed and unparsed entities

Namespace Example

```
<ns:name1
   xmlns:ns="http://www.example.com">
   <ns:name2 ref="ns:name4"/>
   <ns:name3> ns:name5 </ns:name3>
</ns:name1>
```

Entities Example

```
<!DOCTYPE name1 [
  <!ENTITY foo “true”>
  <!ENTITY bar
    SYSTEM “http://www.example.com/bar”>
  <!ATTLIST name1 name2 ENTITY #IMPLIED>
]>

<name1 name2=“bar”> &foo; </name1>
```

RXER

- draft-legg-xed-rxer-xx.txt
- An ASN.1 abstract value corresponds to the ***content*** of an XML element
  - need to provide a root element name
- An RXER encoded LDAP attribute value is a complete XML document
  - root element name is prescribed
  - draft-legg-ldap-transfer-xx.txt

RXER Shortcuts

- Vanilla ASN.1 types don't depend on namespace declarations
  - added for convenience of XML Schema validation
  - must be recognized, but can be ignored
- Comments and processing instructions are dropped
- DTDs in RXER encodings are discouraged

Embedded XML

- draft-legg-xed-glue-xx.txt
- Values of XML schema types are embedded in ASN.1 abstract values using AnyType
- Constraint notation nominates the real type
- AnyType is a SEQUENCE with components for:
  - relevant DTD declarations
  - inherited namespace declarations
  - actual content of an element

AnyType

- AnyType values are self-contained
- AnyType is currently only used for directory attribute syntaxes
  - only apparent in BER and GSER encodings of directory attribute values
  - by default, the LDAP-specific encoding is equivalent to the RXER encoding

Schema Language Strategies

- Representation dimension
  - generated type-specific data structures
  - generic abstract value data structures
  - generic transfer syntax data structures
- Procedural dimension
  - generated type-specific processing routines
  - generic processing routines
    - with compact, optimized in-memory description of types
    - schema checking can be separated from parsing

XML Schema Treatment (1)

- Complications
  - Canonical XML
    - Whitespace, comments and namespace prefixes are significant
    - need to preserve XML Infoset
  - inadequate schema verification
    - broken schemas (invalid restrictions)
  - non-deterministic schemas
    - back-track parsing required

XML Schema Treatment (2)

- Generic transfer syntax data structures approach is least problematic
  - e.g. a realization of XML Infoset
  - less efficient in time and space
  - don’t have to worry about other transfer syntaxes in this case
    - Binary XML will probably not be schema-based

Component Matching

- Component reference notation is insufficient for “components” of XML Schema types
  - XML Schema names can use periods
  - XML Schema allows qualified names
  - embedded ASN.1 values are GSER encoded

Component Paths (1)

- Component path is a generalization of component reference
  - based on XPath syntax
    - uses a different underlying model
    - has extensions for component reference capabilities not expressible by XPath
  - supports qualified names
  - embedded values are RXER encoded
- draft-legg-xed-matching-00.txt

Component Paths (2)

- A component of an ASN.1 type usually has a unique component reference string
  - embedded values are not canonical
- A “component” of an XML Schema type can have many equivalent component paths
  - namespace prefixes are arbitrary

Component Paths (3)

- Component references can usually be compared as octet strings
- Component paths have to be compared at the abstract level
- Component paths are represented as values of AnyType
  - ComponentReference is a UTF8String

Path Assertion

- PathAssertion is an alternative to ComponentAssertion
  - uses a ComponentPath instead of a ComponentReference
- New alternative in ComponentFilter
- RXER encoding is recommended for ComponentFilters with path assertions

PathAssertion Matching

- Need to be able to access the ***content*** of an XML attribute or element for comparison
- The context is significant in a comparison and must also be available
- An XML attribute value can be compared to element character data

User Defined Attribute Syntaxes

- XED allows runtime configurable user defined syntaxes
  - ASN.1, XML Schema, RELAX NG or DTD
  - draft-legg-xed-schema-xx.txt
- XED framework makes the capability available through LDAP
  - Automatically inherited by XLDAP
- Favours generic processing routines and generic data structures

XLDAP (1)

- draft-legg-xed-protocols-xx.txt
- DXER applied to LDAP is unappealing
  - directory data appears as hexadecimal
- Need to remediate the LDAP ASN.1 specification
  - OCTET STRINGs revert to original X.500 definition

XLDAP (2)

- Protocol message and directory data are uniformly encoded in XML
- Directory attribute values can no longer have self-contained context
- Namespace differences can be taken care of with local namespace declarations

XLDAP (3)

- Notation and unparsed entity declarations must be collected at the beginning of the operation encoding
  - particularly bothersome in X.500 protocols
- Entity and notation names are not globally unique
  - names may need to be remapped

Conclusion

- draft-legg-xed-roadmap-xx.txt
- XED mailing list: xeddev@adacel.com
  - soon to be xeddev@eb2bcom.com
- XED web site: www.xmled.info