Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

ISO/IEC 19757-9 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information Technology*, Subcommittee SC 34, Document Description and Processing Languages.

— *Part 1: Overview*

— *Part 2: Regular grammar-based validation — RELAX NG*

— *Part 3: Rule-based validation — Schematron*

— *Part 4: Namespace-based validation dispatching language — NVDL*

— *Part 5: Datatype Library Language — DTLL*

— *Part 6: Path-based integrity constraints*

— *Part 7: Character Reportoire Description Language — CRDL*

— *Part 8: Document Schema Renaming Language — DSRL*

— *Part 9: Datatype- and namespace-aware DTDs*

— *Part 10: Validation Management*

Introduction

The language of Document Type Definitions (DTDs) was the original schema language defined by W3C XML and was closely based upon the DTD language defined by SGML. For a variety of reasons, both technical and economic, many users of XML for document-centric applications, especially among those who were previously (and in some cases continue to be) users of SGML, still favour the use of DTD language for grammar-based schema definition in such applications.

It is important to provide users that have made a significant investment in DTDs with a migration path that will enable them to adopt DSDL without having to translate all their existing DTDs to a different schema language, especially as this would oblige them to replace all systems that only work with DTDs, with all the expense and organizational upheavals thereby entailed. A sensible migration path should enable such users to continue to use DTDs for as much of the document validation process as can reasonably be managed, but also enable them to reap the benefits of using those parts of DSDL that most obviously complement and extend the use of DTDs for validation purposes.

It is equally important that the migration path should enable users to continue to use legacy systems that are incapable of using any kind of extension to the DTD language, while at the same time introducing new systems that are equipped to use such extensions. The method of extension must therefore be such that DTDs with extended functionality are valid XML DTDs in accordance with W3C XML.

The most significant validation tasks that cannot be performed using a DTD alone are:

— validation of names with respect to namespaces

— validation of data content and attribute values with respect to datatypes

— rules-based validation

Of these three, the last is made possible by implementation of Part 3 of DSDL. The first two tasks are currently only possible if Part 2 of DSDL is implemented, but existing DTD users are unlikely to wish to maintain parallel RELAX NG and DTD schemas for each application simply as a means of supporting the use of namespaces and datatypes.

For these reasons this International Standard addresses the extension of DTDs to support the use of namespaces and datatypes.

Document Schema Definition Languages (DSDL) — Part 9: Namespace- and datatype-aware DTDs

1 Scope

This International Standard defines a language that is designed to extend the functionality of an XML DTD to include:

— specifying one or more namespaces to which some or all of the element and attribute names in a DTD belong

— constraining elements with content model 'ANY' to contain elements whose names belong to one or more specified namespaces

— specifying datatypes for elements that contain data content and for attribute values.

Two alternative syntax bindings for this language are defined. One binding uses the syntax of XML processing instructions and is designed to enable declarations in this language to be embedded within an XML DTD whose functionality is to be extended, or within an XML instance to be validated against such a DTD, without invalidating either DTD or instance so far as legacy parsers are concerned. This syntax is defined in Clause 4 using the modified BNF syntax notation used in W3C XML.

The second syntax binding is XML-based and is defined in Clause 5. The syntax rules are defined by a schema that conforms to the RELAX NG Compact Syntax defined in Part 2 of this standard. This syntax is designed to enable declarations in this language to be expressed in XML, to facilitate implementation using existing XML tools, either as a namespace-qualified fragment embedded within an XML instance or as a separate XML document.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 19757. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO/IEC 19757 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

SGML, *ISO 8879:1986 Information Processing - Text and Office Systems - Standard Generalized Markup Language (SGML)*, http://www.iso.org/

W3C XML, *Extensible Markup Language (XML) 1.0 (Third Edition)*, W3C Recommendation, 30 October 2003, http://www.w3.org/TR/2003/PER-xml-20031030/

W3C XML-Names, *Namespaces in XML*, W3C Recommendation, 14 January 1999, http://www.w3.org/TR/1999/REC-xml-names-19990114/

RELAX NG, *Grammar-based schema language (RELAX NG)*, ISO/IEC 19757 Part 2, http://www.dsdl.org/

DTLL, *Datatype Library Language*, ISO/IEC 19757 Part 5, http://www.dsdl.org/

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply:

3.1 qualified datatype name

3.2 unqualified datatype name

4 DTD extension declarations using XML processing instructions

4.1 Syntax using XML processing instructions

The following set of syntax productions define how namespace and datatype declarations expressed in accordance with this standard may be represented as XML processing instructions. Syntax productions for *name-token-group*, *NCName*, *URI*, *lit*, *lita* and *s* are given in W3C XML.

DTD-extension-processing-instruction ::= *pio* "DSDL-9" *s* (*namespace-prefix-binding-declaration* | *namespace-name-binding-declaration* | *wildcard-namespace-qualifier-declaration* | *default-datatype-library-declaration* | *datatype-library-prefix-binding-declaration* | *datatype-declaration*) *pic*

namespace-prefix-binding-declaration ::= "namespace-prefix-binding" *s* "namespace-URI=" *URI-literal s* "prefix=" *prefix-literal*

namespace-name-binding-declaration ::= "namespace-name-binding" *s* "namespace-URI=" *URI-literal s* "applies-to=" *name-locator-literal*

wildcard-namespace-qualifier-declaration ::= "wildcard-namespace-qualifier" *s* "wildcard-namespace-list=" *wildcard-namespace-literal s* "applies-to=" *name-locator-literal*

wildcard-namespace-literal ::= ((*lit wildcard-namespace-list lit*) | (*lita wildcard-namespace-list lita*))

wildcard-namespace-list ::= *URI* (*s URI*)*

default-datatype-library-declaration ::= "default-datatype-library=" *URI-literal*

datatype-library-prefix-binding-declaration ::= "datatype-library-prefix-binding" *s* "datatype-library-URI=" *URI-literal s* "prefix=" *prefix-literal*

datatype-declaration ::= "datatype=" (*datatype-name* | *datatype-qname*) *s* "applies-to=" *name-locator-literal*

URI-literal ::= ((*lit URI lit*) | (*lita URI lita*))

name-locator-literal ::= ((*lit name-locator-sequence lit*) | (*lita name-locator-sequence lita*))

name-locator-sequence ::= (*s*)? *name-locator* (*s name-locator*)*(*s*)?

name-locator ::= ("element" *s* (*name-token-group* | "*")) | ("attribute" *s* (*name-token-group* | "*") *s* of-element *s* (*name-token-group* | "*"))

datatype-qname ::= ((*lit prefix* ":" *datatype-name lit*) | (*lita prefix* ":" *datatype-name lita*))

prefix-literal ::= ((*lit prefix lit*) | (*lita prefix lita*))

prefix ::= *NCName*

datatype-name ::= *NCName*

4.2 Examples of declarations using processing instructions

EXAMPLES TO BE INSERTED HERE - ED

5 XML syntax

The following schema in RELAX NG Compact Syntax specifies how a set of namespace and datatype declarations expressed in accordance with this standard may be represented as an XML document.

```
datatypes xsd = "http://www.w3.org/2001/XMLSchema-datatypes"
            default namespace = "http://dsdl.org/dsdl-9"

            start = document

            document = element dtd-extension { head, body }

            head = (element applies-to-dtd {public | system} )*

            public = element public {text}, system?

            system = element system {xsd:anyURI}

            body = ( namespace-prefix-binding |
            namespace-name-binding |
            wildcard-namespace-qualifier |
            default-datatype-library |
            datatype-library-prefix-binding |
            datatype )+

            namespace-prefix-binding = element namespace-prefix-binding {
            element namespace-uri {xsd:anyURI},
            element prefix {xsd:NMTOKEN}
            }

            namespace-name-binding = element namespace-name-binding {
            element namespace-uri {xsd:anyURI},
            element applies-to {(element-type-names | attribute-names)+}
            }

            element-type-names = element element-type-names {(name+ | any)}

            attribute-names = element attribute-names {
            (name+ | any), element-type-names?
            }

            name = element name {xsd:NMTOKEN}

            any = element any {empty}

            wildcard-namespace-qualifier = element wildcard-namespace-qualifier {
            (element namespace-uri {xsd:anyURI})+,
            element applies-to {element-type-names}
            }

            default-datatype-library = element default-datatype-library {
            element library-uri {xsd:anyURI}
            }

            datatype-library-prefix-binding
            = element datatype-library-prefix-binding {
            element library-uri {xsd:anyURI},
            element prefix {xsd:NMTOKEN}
            }

            datatype = element datatype {name,
            element applies-to {(element-type-names | attribute-names)}
            }
```

6 Inclusion of namespace information in DTDs

If this standard is used to associate names used in a DTD with namespaces, a name in the DTD may only contain a colon ":" if that colon separates qualifying prefix and local name in the manner described in W3C XML-Names.

A namespace URI may be associated with specified element type name(s) or attribute name(s) that include qualifying prefixes, by declaring a corresponding namespace prefix binding.

Each prefix may only be bound once to a namespace URI. If two or more namespace prefix binding declarations specify the same prefix, the second and subsequent declarations are ignored.

A namespace URI may be associated with specified element type name(s) or attribute name(s) that do not include qualifying prefixes, by declaring a corresponding namespace name binding.

An asterisk following the type of name 'element' indicates that the declaration applies to all unqualified element names. An asterisk following the keyword 'attribute' in a name locator literal indicates that the declaration applies to all unqualified names of attributes on the indicated elements.

An element name, or an attribute name of a given element, may only be declared to be in a single namespace. If two or more namespace name binding declarations specify the same element name, or the same attribute of the same element, the second and subesequent bindings are ignored.

An unqualified name has no namespace unless a namespace name binding is declared that applies to this name.

A namespace URI may be associated with the content of an element type whose content model is declared to be "ANY", by defining a wildcard namespace qualifier declaration for that element type. A namespace URI may only be specified in a wildcard namespace qualifier declaration if it occurs either in a namespace prefix binding declaration or in a namespace name binding declaration for the same DTD.

7 Inclusion of datatype information in DTDs

A datatype may be associated with a specified element type name or attribute name, by including a corresponding datatype declaration in the DTD.

Every datatype name must be associated with a declared datatype library. If a specified datatype name includes a qualifying prefix, a corresponding datatype library prefix binding declaration must be included in the DTD.

Each qualifying prefix may only be bound once to a datatype library. If two or more datatype library prefix binding declarations specify the same prefix, the second and subsequent declarations are ignored.

If a datatype name is specified without a qualifying prefix, the DTD must include a default datatype library declaration. A DTD may only have one default datatype library. The second and subsequent default datatype library declarations are ignored.

The URI specified in either a datatype library prefix binding declaration or a default datatype library declaration must identify a datatype library, for example a library that conforms to Part 5 of this International Standard (DTLL).

If two or more datatype declarations apply to the same element content or attribute value, the second and subsequent declarations are ignored.