



**XML DATA VALIDATION STANDARDS  
VERSION 1.0**

**JUNE 22, 2007**

---

**Information and Technology Management Branch**

**❖ IM / IT Standards & Guidelines ❖**

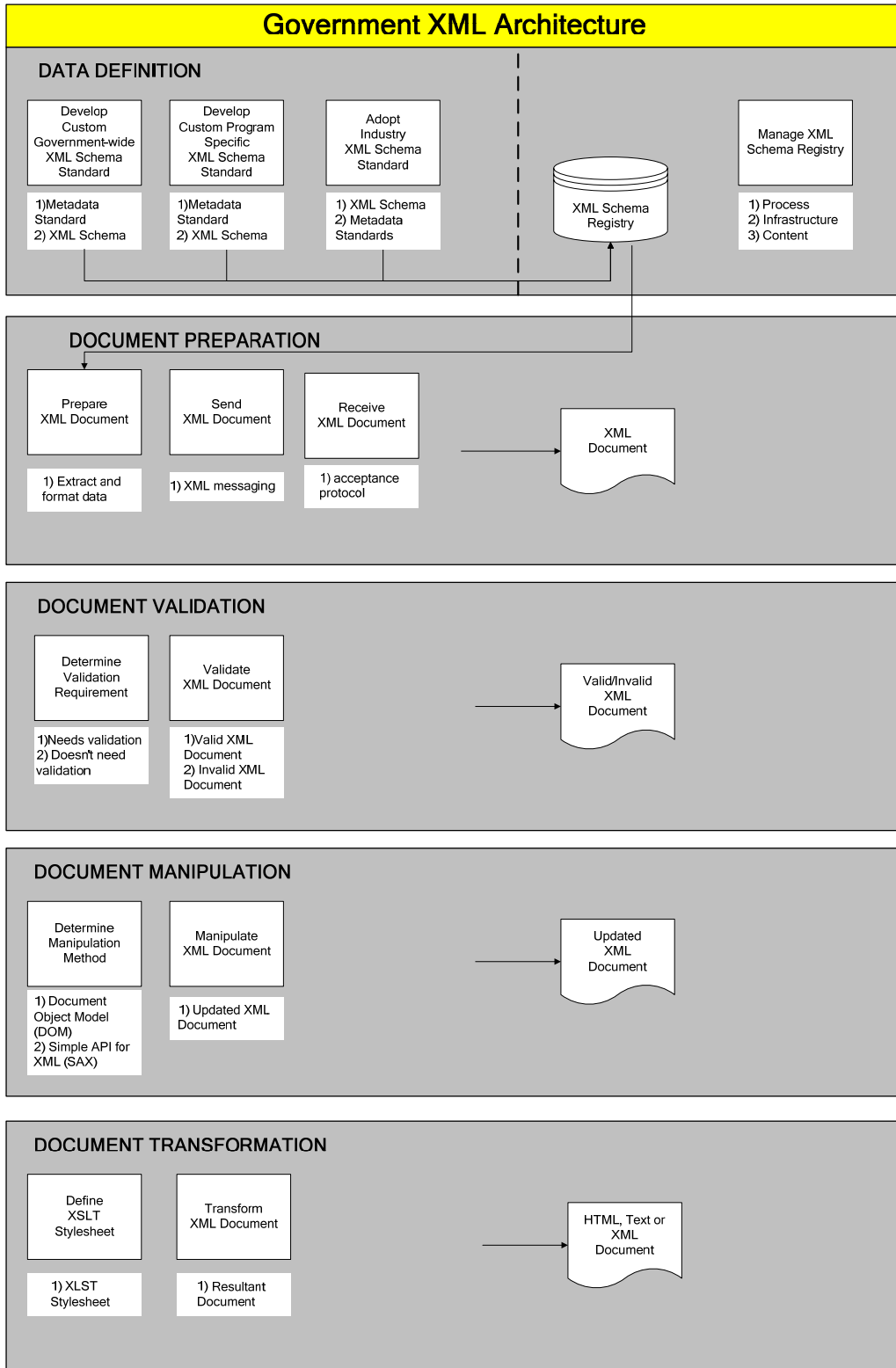
<b>1. XML STANDARDS CONTEXT .....</b>	<b>3</b>
<b>2. DOCUMENT PURPOSE.....</b>	<b>5</b>
<b>3. XML DATA VALIDATION.....</b>	<b>5</b>
<b>4. XML SCHEMA DEFINITION (XSD).....</b>	<b>6</b>
<b>5. DOCUMENT OBJECT MODEL (DOM) .....</b>	<b>8</b>
<b>6. SIMPLE API FOR XML (SAX).....</b>	<b>8</b>

## 1. XML Standards context

XML Technologies span a wide variety of areas and standards. Ministry ADE process has identified the following areas of XML where standards could be built and evolved. Standards for each area are covered in separate documents.

- *XML Core Standards* : [“well-formed-ness” standards]
- *XML Data Validation Standards* : [XML Schema standards] : **This document.**
- *XML Namespaces* : [standards on resolution of tags with similar tag names]
- *XLink and XPointer* : [standards for creating hyperlinks in XML documents and standards for hyperlinks to point to more specific parts (fragments) in the XML document]
- *XML Catalogs* : [URL redirection standards]
- *XQuery* : [XML data querying standards]
- *XPath* : [standards on navigation through elements and attributes in an XML document]
- *XSL-FO (also known as XSL)* : [standards on formatting of XML data for output]
- *XSLT* : [standards on transforming XML documents]

The following diagram depicts the Government XML Architecture at a high level. This architecture is evolving and subject to changes :



## 2. Document Purpose

The purpose of this document is to provide a set of standards to ensure data contained in XML documents is of appropriate quality, i.e. ‘valid’. The document assumes that the reader has prior knowledge of XML. This document does not describe syntax validation known as “well-formedness” standards as these are described in “XML Core standards” document. This document also does not describe the data definitions which are used to validate data in XML documents.

## 3. XML Data Validation

A *well-formed* XML document only ensures that it complies with all the syntax rules and thus will not generate any parsing errors. However, *well-formed-ness* does not guarantee the validity of data in the XML document - it only ensures the syntax of the XML document is acceptable. To ensure that XML data is valid, the XML document needs to be validated against a set of rules called **Grammar**. The use of **DTD** or **XML Schema Definition** makes the XML data self-validating against a set of rules. Therefore, an approved DTD or XML Schema Definition is mandatory pre-requisite to being able to validate data in XML documents.

Data validation must be applied in two areas: data structure and data content. Data structure is defined by the XML Schema Definition. An XML document must completely match the defined data structure to be considered valid, otherwise it is considered ‘rejected’ or invalid. Where an approved list of data values for a data element is defined in the XML Schema Definition, such as acceptable gender codes or valid date ranges, data in the XML document must adhere to the data value constraints to be valid, otherwise, the XML document is ‘rejected’ or invalid. Data value constraints govern data content.

An XML document cannot be partially valid. It must be wholly accepted as valid or rejected as invalid.

An XML document that is valid in data structure and valid in data content is considered completely valid.

**Ministry Application Development Environment (ADE) requires that in addition to “well-formed-ness”, all XML documents must also be completely “valid”.**

**DTD versus XSD (XML Schema) : Ministry ADE requires the use of the XSD (*XML Schema*) version 1.1 Standard for XML data validation. Document Type Definition (DTD) standards that preceded XML Schema standards, must not be used.**

**Please follow the XML Schema standards defined in subsequent sections while authoring/generating the XSD files.**

## 4. XML Schema Definition (XSD)

The Ministry can develop custom XML Schema definitions for some data subject areas, but other areas are governed by Education Sector, industry and central government XML Schema standards to ensure data interoperability. The following are the various components/features/syntax of XML Schema Definition standard for custom developed schemas. These strive to adhere to broader standards and practices. However, where there is conflict with broader standards, exceptions will be granted and the broader standard should be followed.

The following are based on the open W3C XML Schema Version 1.1 standard:

1. Each XML Schema definition must be derived from, and adhere to an approved logical data model. The logical data model standards are definition in the [Requirements Modeling Standards and Guidelines document](#).
2. All XML Schema documents are (and must be) well-formed XML documents.
3. The root element in a XML Schema document is “*schema*” and belongs to the schema name space (<http://www.w3.org/2001/XMLSchema>).
4. XML Schema namespace must be imported in order to use the XML Schema tags.
5. The namespace URI is always the same as the XML Schema namespace, but the local name is arbitrary.
6. The **xsd** or **xs** are commonly used as prefixes for XML Schema namespace. However, the Ministry ADE requires the “**xsd**” prefix.
7. XML Schema documents (XSD documents) contain a set of rules by which XML instance documents are validated.
8. The main contents of XML Schema document are : Element and Attribute declarations, Data type definitions for pre-defined and custom defined data types etc.
9. *Data types* in XML Schema must be either : Simple, Complex, Atomic, List, Union, Built-in, Primitive and Derived.
10. *Model groups* define the order in which elements appear within a complex type. Model groups in XML Schema are defined using the following tags :

**xsd:sequence**  
**xsd:choice**  
**xsd:all**

11. Occurs tags : XML Schema offers facilities to enforce constraints on number of occurrences of a component through “occurs tags”. The “occurs tags” can be attached to elements as well as model groups. The two occurrence tags are :

**xsd:minOccurs** - indicates the minimum number of times a component can occur  
**xsd:maxOccurs** - indicates the maximum number of times a component can occur

12. Defining Attributes in XML Schema : Only complex types can have attributes and simple types cannot have attributes. XML Schema attributes are defined using the “*xsd:attribute tag*” which has the following syntax :

```
<xsd:attribute name="attribute-name" type="attribute-data-type" use="required" or  
"optional" or "prohibited" fixed="constant value" default="default value"/>
```

where

“name” indicates the name of the attribute. This element is mandatory.

“type” indicates the data type of the attribute.

Value must be Simple, Complex, Atomic, List, Union, Built-in, Primitive or Derived. This element is mandatory.

“use” indicates if the attribute is required, optional or prohibited. Values must be ‘required’, ‘optional’, ‘prohibited’. Note the values are in lower case. This element is mandatory.

“fixed” indicates the constant value of the attribute. This element is optional. The constant value must be consistent with the format of the data element as defined in the Logical Data Model.

“default” indicates the default value of the attribute. This element is optional.

The name of an attribute must follow the same naming conventions defined for logical data models, as defined in the Requirements Modeling Standards and Guidelines.

13. Annotations in XML Schema : The annotation element is a top level element that specifies schema comments. The comments serve as inline documentation. In addition to the normal XML comments, the XML Schema provides the “xsd:annotation” tag for documentation. Annotation tags can occur anywhere in the document and can contain two child tags :

“xsd:documentation” : meant to contain any human readable content

“xsd:appInfo” : meant to contain only machine readable content

14. Associating XML Schema with XML document : To associate an XML document with a schema, two schema attributes are available :

“xsi:schemaLocation” = “namespace-URI schema-doc-URI”, which is for XML tags that are namespace qualified

“xsi:noNamespaceSchemaLocation” = “schema-doc-URI”, which is for XML tags that do not use a namespace

In both cases, the “schema-doc-URI” represents the location of the schema file either on the local file system or on web server.

15. Identity Constraints in XML Schema : The XML Schema tags as shown below :

“xsd:unique” : The unique element defines that an element or an attribute value must be unique within the scope.

“xsd:key” : The key element defines that an element or an attribute has a value and the value is unique.

“xsd:keyref” : The key element defines that the values point to an existing xsd:key values (similar to foreign key constraint in databases).

The use of these tags must be consistent with the logical data model.

## 5. Document Object Model (DOM)

Document Object Model (DOM) is a W3C specification for accessing and manipulating XML documents in a standard and consistent manner. DOM specifications are currently at Level 3 officially. DOM based parsers create an internal tree based on the hierarchical structure of the XML data. The tree can be navigated and manipulated from the application and it stays in memory until it is released. DOM uses functions that return parent and child nodes, giving full access to the XML data and providing the ability to interrogate and manipulate these nodes. The DOM is designed to be used with any programming language. DOM is a good choice when the document is not too large. If the document (and the tree generated by DOM) is large, using DOM can put a strain on system resources since the DOM tree resides in the system memory.

## 6. Simple API for XML (SAX)

Simple API for XML (SAX) is a public domain API developed cooperatively by the members of the XML-DEV (XML DEVELOPMENT) Internet discussion group. SAX API are currently at version 2.0.1. Although SAX was originally a Java-based API, it is now available for other programming languages too. SAX is a publicly developed standard for the events-based parsing of XML documents. SAX defines an abstract programmatic interface that models the XML information set (infoset) through a linear sequence of familiar method calls. SAX facilitates the search of large documents to extract small pieces of information and allows aborting of processing after the information is located. SAX does not demand resources for an in-memory representation of the document.

**DOM versus SAX : Ministry ADE process recommends and supports SAX approach by default. Ministry ADE process recommends that DOM approach should be used only when there is strong business case and technical case for DOM usage.**

**SAX can also be used to build DOM trees (or portions of DOM trees); conversely, developers can also traverse DOM trees and emit SAX streams. Ministry ADE process supports the usage of DOM and SAX together if there is strong business case and technical case for such a situation.**