4 Using the SDP XML Schema for XML Document Validation

4.1 Introduction

The Transit Schedule Data Exchange Architecture (TSDEA) project describes the exchange requirements for schedule and related data. These data requirements are incorporated into Schedule Data Profile (SDP) reference data model and implemented into the SDP XML Schema.

This tutorial provides a brief overview of SDP XML Schema elements, and then describes two applications that use the SDP XML Schema to validate XML documents. A thorough treatment of the XML schema implementation and conceptual data reference model is contained in Chapter 2 of *SDP Guidance Documentation Part 2: User Requirements*.

One of the advantages of XML is the relative ease of verification of conformance with an XML schema specification. Furthermore, software tools that are easy to use are readily available. Two applications that validate XML documents are discussed in this chapter. Finally, XML, which is written in ASCII, is portable across system platforms.

One disadvantage of using XML documents is the relative large size of the document, routinely ranging between 5 to 15 megabytes in size, based on a representative sample of SDP XML documents.

References:

SDP Guidance Documentation Part 2: User Requirements Version 1.0, June 2008

The intended audience f this manual includes:

- System developers and data modelers interested in creating valid SDP XML data; and
- System managers responsible for setting up the run-time environment for SDP applications

4.2 Structure of the SDP XML Schema

The SDP XML Schema is comprised of four files. These are listed below:

- SDP_XML_Schema_v1_0.xsd: This is the base schema, which imports the SDP_Common and SDP_Domain schemas. It defines the structure of the SDP at the highest level. The SDP100 root element contains the following:
  - AgencyRegistration
  - Service
  - TransitNetwork
  - TransitGazatteer
  - TransitFacilities

- SDP_common_v1_0.xsd – Include a definition of complex elements included by the highest level elements of the SDP.
- SDP_domain_v1_0.xsd – Includes a definition of identifiers and code enumerations used in the SDP.
- GML_geometry.xsd - Contains definitions of the Geography Markup Language (GML) used in the SDP.

4.3 Structure of the Schedule Calendar Date (SCD) XML Schema

A second schema that is part of the demonstration project is the Schedule Calendar Date schema, which is made up of two files. These are listed below:

- SDP_Schedule_Calendar_Date_v1.xsd: This is the base schema, which imports the SDP_domain_scd schema. It defines the structure of the Calendar and Schedule_Calendar_Date. The Calendar root is comprised of Schedule_Calendar_Date elements.
- SDP_domain_scd_V1_0.xsd – Includes a definition of identifiers and code enumerations used in the SCD.

4.4 Elements of the XML Schema

The SDP Reference Data Model is an implementation neutral representation of the static design that fulfills the SDP user requirements. The SDP data model describes the static data structure (data relationships and valid value rules) for the information used in SDP applications. The SDP data model specifies the following:

- Sequence and order of data
- Multiplicity, or number of times, a data element can be represented
- Defines re-usable types (for example, longitude and latitude are used together, so a re-usable point type may be defined)
- Specifies mandatory (1 of more required) and optional (0 or more required) data elements
- Specifies data value ranges and other constraints (for example, only the floating point numbers -90.000000 through 90.000000 may be used to define a latitude value, only the following valid text codes may be used to describe a dayType: "weekday", "mon", "tue", …).
- Key References

4.5 Validating XML Documents Using the XML Schema

The SDP demonstration project used the Altova XMLSpy software and a Microsoft Windows Scripting Host application developed as part of the demonstration project to validate both the XML schemas and XML documents. How to use these two applications to validate XML documents are the topics of Section 4.6 and Section 4.7 below.

4.6 Using Altova XMLSpy

Figure 1 on the following page shows XMLSpy after startup of the sdp.spp, an XMLSpy project file. The XMLSpy project file contains a list of schema files and XML documents. A portion of the sdp_MNR_Base_April2008.xml file is shown in the center.

At the bottom of the figure, and part of the XMLSpy application, is a panel containing a Validation Report that indicates that the sdp_MNR_Base_April2008.xml document is valid. This means that the XML document conforms to all the schema rules including:

- Sequence and order of elements
- All mandatory elements present
- All data content within the value constraints

If we change one of the data values to an incorrect value, for example the effectiveDate attribute of the <Agency> element to 2008-13-01, and run the validation, the XMLSpy application will provide a description of the error in the Validation Report panel at bottom, highlight the location of the error in the document, and print the xPath location of the error found. This is shown in Figure 2.

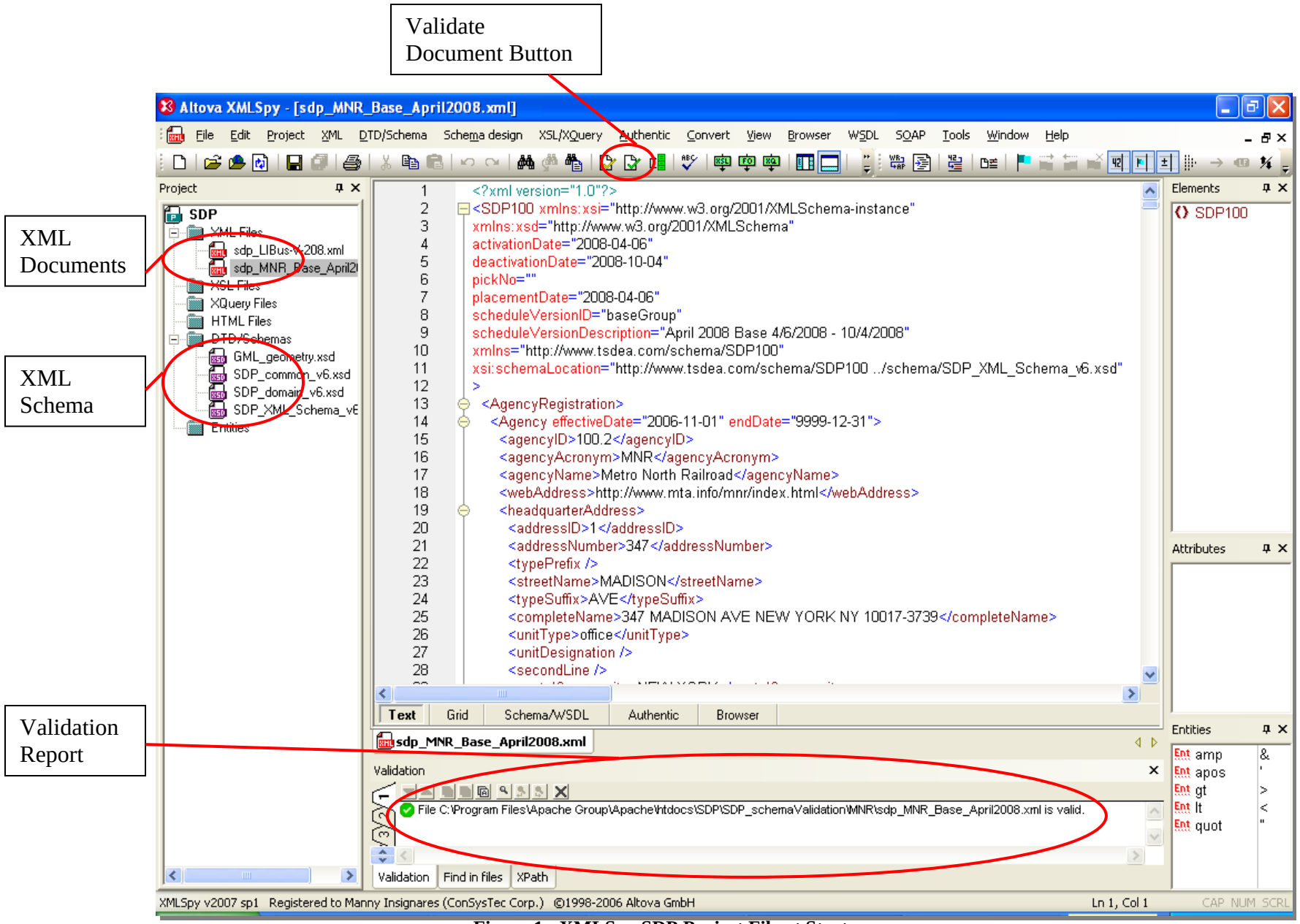
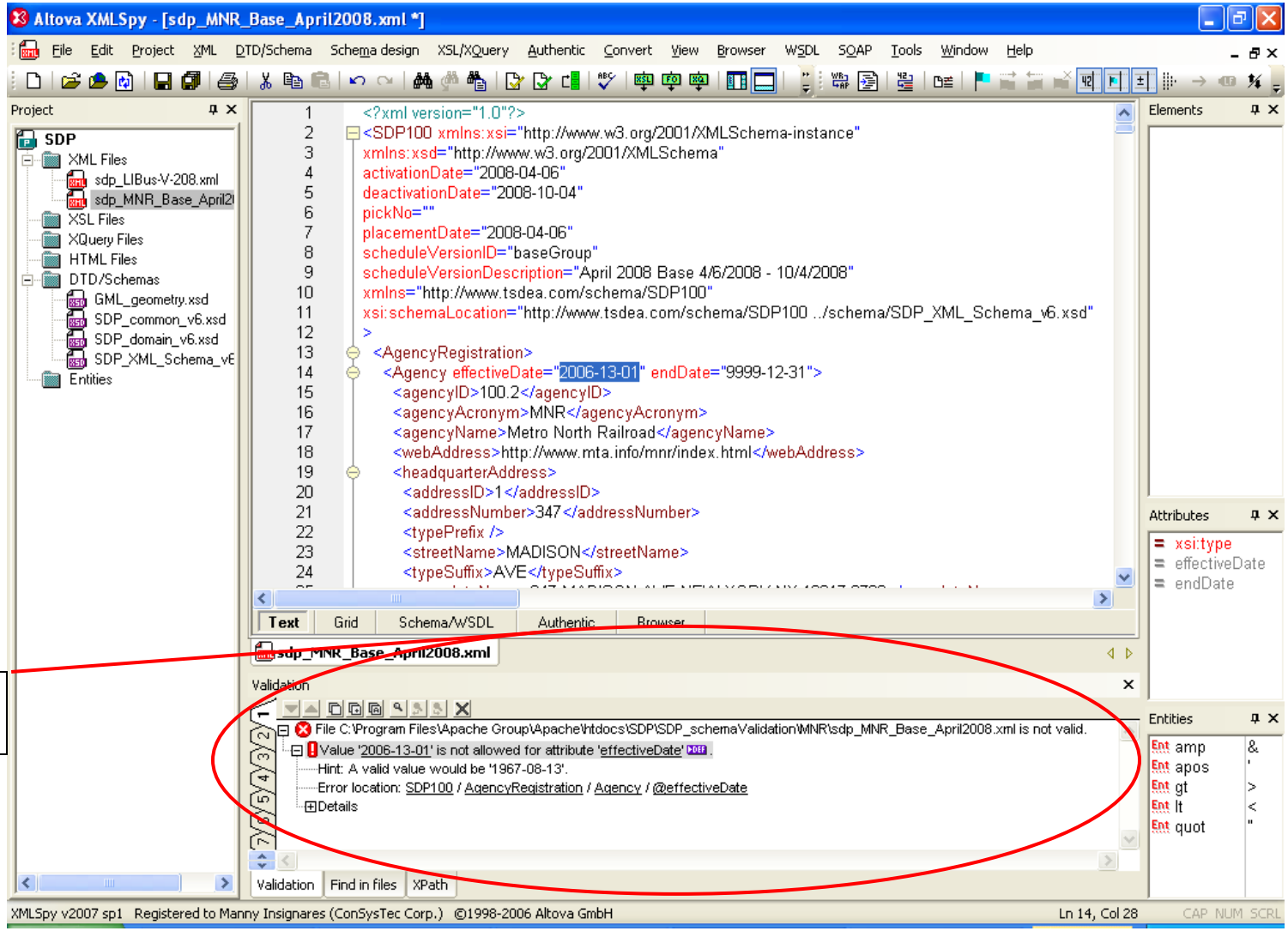


Figure 1: XMLSpy SDP Project File at Startup



Validation Error Report

Figure 2: XMLSpy Showing Validation Errors

4.7 Microsoft MSXML2 Windows Scripting Host Application

4.7.1 Software Installation

A sample application, msxsd, based on the MSXML2, an XML processor dynamic link library (DLL) that comes with the Microsoft Internet Explorer application is included on the application CD. The msxsd is a javascript application what accesses the MSXML2 DLL to validate an XML document against an XML Schema. The table below outlines the operating system and version requirements for the database instance example.

System	Version
SDP XML Schema	Version 1_0
Windows Scripting Host	5.6
Microsoft XML DLL	2
MS Windows	Windows XP

Prerequisites: Microsoft Windows Scripting Host.

The attached CD contains a copy of the javascript, but not the Windows Scripting Host software nor the MSXML2 DLL.

Appendix A of this chapter contains a copy of the javascript source and batch file to execute the code under the Windows Scripting Host.

4.7.2 Application Execution

To start the msxsd application, navigate to the $SDP/SDP_schemaValidation/MNR directory. Then, double-click on the file called validate_MNR_SdpXml.bat to start the application.

If the file is valid, then the following prompt will display (see Figure 3):

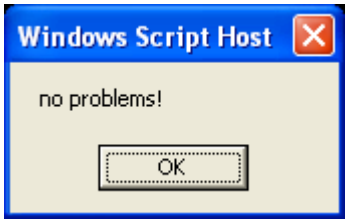


Figure 3: MSXSD Application Showing a Validation Report

An example showing an error report message is in Figure 4:

Figure 4: MSXSD Application Showing Validation Error Report

5 Appendix A: MSXSD.JS Source and Batch Files

File: msxsd.js

Validate XML Document Against Schema

```
// validate parameters
if(WScript.Arguments.length != 3) {
    WScript.Echo("msxsd takes three arguments - datafile, namespace, schema - eg:");
    WScript.Echo('msxsd books.xml "" books.xsd');
} else {
    var cache = new ActiveXObject("Msxml2.XMLSchemaCache.4.0");
    cache.add(WScript.Arguments(1), WScript.Arguments(2));

    var xmldoc = new ActiveXObject("Msxml2.DOMDocument.4.0");
    xmldoc.async = false;
    xmldoc.preserveWhiteSpace = true;
    xmldoc.schemas = cache;
    xmldoc.load(WScript.Arguments(0));

    if(xmldoc.parseError.errorCode != 0)
        WScript.Echo("There is a problem: " + xmldoc.parseError.errorCode + " " +
xmldoc.parseError.reason);
    else
        WScript.Echo("no problems!");
}
```

File: validate_MNR_SdpXml.bat

```
msxsd SDP_MNR_Base_April2008.xml "http://www.tsdea.com/schema/SDP100"
../schema/SDP_XML_Schema_v1_0.xsd
```