Appendix L

New York State ITS Standards Specification Development Guide

ITS Standards Test Procedure Development Guidance and Tools

Prepared for

New York State Department of Transportation

Prepared by

Consensus Systems Technologies Corp.

January 22, 2006

Table of Contents

Revision History

Filename	Version	Date	Author	Comment
NYStateSpecDevGuide – ApL – Test Procedure Guidance and Tools.doc	0.4	01/22/06	M. Insignares / P. Chan	Initial Draft

1 Introduction

This appendix builds on the discussion on dialog-based test procedure development. The crux of the discussion is that a dialog-based testing process (requirements test cases to test plan to test procedures to implementation test) allows dialogs defined at the requirements stage of design to carry through to implementation test to provide traceability throughout the design process. The Dialog Worksheet and RTM play a critical role in this traceability.

This section expands the discussion and provides guidelines for ITS standards testing at all levels of the ITS communication framework (information to subnetwork level). All levels must be specified and work together for an implementation and moreover all must be tested – no easy feat given the relative immaturity of ITS standards deployment experience and test tool availability.

A guiding principle of this appendix is that TCP and UDP/IP marry software in devices and center-based systems to the communications infrastructure.

Finally, test procedures without test tools would prove problematic and a preliminary assessment of test tool capabilities is included. Thus, specifications have worthy and meaningful test procedures that in turn can be proved through test tools.

2 Test Procedures and Tools for Center-to-Field Communications

2.1 Customize and Verify Dialogs and Interface / Object Content

The test procedure guidance provided here assumes that dialogs for center-to-field communications have been created and are based on project requirements. Moreover, that a PRL / PICS and Requirements Traceability Matrix have been created.

The RTM ties a solution (dialogs, interfaces, and objects) to the project requirements. For field communications the RTM represents a detailed dialog worksheet that can be extended to facilitate test procedure development.

2.2 Develop and Verify Test Data

It is recommended that a customized projects-specific MIB be created that provides a computer-readable form of the object definitions. The object definitions are just that – definitions – they are note what is sent across a wire. Object definitions for example define value ranges and value formats for objects, but are not the values themselves. Therefore, test data must be created in order to test that an interface and objects have been properly implemented. It is recommended that the test data be written in MIB format (text representation), called a MIB Instance, and that the MIB Instance be validated against the National and Project-specific MIB. In this way the test data can be proved to conform with the National and Project-specific MIB.

2.3 Center-to-Field Communications Test Tools

The table that follows contains a summary of test tools and capabilities as relates to center-to-field communications. The table is organized to show tools as apply to the ITS Communications Framework levels.

NOTE: The information in the table below is NOT based on actual use and evaluation of the test tools – it is, rather, a preliminary assessment of test tools based on vendor cut-sheets, vendor web site information, the personal experience of the authors, and reports that relate to NTCIP test tool evaluation by others.

Table 2-1. Role of Test Tool as Applies to ITS Center-to-Field Communications

	Information Level NTCIP 1200 Series			Application Profile NTCIP 2301			Transport Level NTCIP 2201		SubNet TCP-UDP/IP	Transp Level NTCIP 2202	SubNetwork TP-NULL		
Tool	**MIB Verification**	**Dialog Pattern / Content Verification**	**Content Verification against MIB (1)**	**SNMP - BER**	**SFMP-OER (2)**	**STMP – OER (2) (3)**	**TCP / UDP**	**IP**	**Ethernet (802.11) NTCIP NTCIP 2104**	**TP-NULL (4)**	**PPP NTCIP 21XX**	**PMPP/ FSK NTCIP 21XX (5)**	**PMPP/ RS-232 NTCIP 21XX**
SMIC	X												
SNMP Client		X		X									
SimpleSoft NTCIP		X		X									X
FTS NTCIP		X		X									X
NTester		X		X									
IDI		X		X		X							
TTCP							X						
PING								X					
PPP Dialer											X		
Ethernet Packet Sniffer									X				

Known Issues:

(1) It is unclear that there are tools to verify byte encoded content transmitted between a SNMP Management Station and SNMP Agent against a text MIB. One approach to resolving this issue is to create a text MIB Instance (a textual representation in MIB format of the information that is to be transmitted). The MIB Instance can be compiled into BER or OER byte encoding and saved to a file, creating a byte template for comparing wire transmissions. Thus, a text MIB Instance could be created and validated against a National or Project-specific MIB, the MIB Instance used to create a byte encoded BER or OER template, which could be used for verification of byte encoded information against a given MIB.

(2) It is unclear that there are tool to verify proper OER encoding, and no tools known to verify SFMP.

(3) STMP is only required for ASC (Actuated Signal Controllers).

(4) It is unclear that there are any tools that will verify TP-NULL

(5) It is unclear that there are any tools that will verify TP-NULL / PMPP / FSK communications.

2.4 Center-to-Field Communications Test Procedure Guidance

2.4.1 Information Level Test Procedures

2.4.1.1 Verify the National MIBs

It is important to ensure that all National MIBs referenced by the project are available and verified both in their own right and that they are verified to work together – in other words, there are no unreferenced or unresolved elements. This is not a task that has been handled by the National Standards as it is not possible to pre-determine all combinations that might be implemented in projects.

Importantly, the MIBs must be verified with a tool that will be specified prior to testing.

2.4.1.2 Verify the Project MIBs

In this activity the customized MIBs are verified, much as outlined in section 2.4.11.

NOTE: There is likely no way to verify that a subset MIB (the project's MIB) is valid against the National MIB. Therefore, to prove conformance all information level content must be validated against both MIBs.

2.4.1.3 Verification of Dialogs

The following assumptions are pre-conditions to information level dialog testing:

- National and Project MIBs have been verified
- Project-specific PICS, RTM, and MIB include all mandatory elements of the national standard, and therefore these project-specific documents will ensure conformance with the national standards
- Test data have been created and verified
- Dialogs to be tested are contained in the RTM

Dialogs are implemented for center-to-field communications as a sequence of GET and SET actions. Pre-conditions for dialogs generally include testing certain objects are "pre-set" and correct (these pre-sets are dialog specific). In addition to the general procedures shown in the table before, it is useful to include columns for expected and actual results (these are not shown and would be dialog specific).

Table 2-2. Center-to-Field Communications Dialog Test Procedures (Generalized)

Test Pattern ID	Dialog Pattern	Encoding /Protocol	Tools	Pre-Conditions	Procedure
C2F-1	Simple Get	BER / SNMP	SNMP Client	Verify Pre-Set Values. Prepare MIB-verified test GET-Request data. Prepare BER MIB Instance Templates (valid byte encoded responses)	1. Do GET-Request using prepared BER encoded request (verifies protocol and end-point) 2. Save BER Encoded Stream (GET-response) to a File OR Write Down Byte Codes for Verification 3. Verify Saved BER File (OR bytes) is BER (encoding is BER) 4. Verify Saved BER File (OR bytes) against BER encoded MIB Instance template (verifies correct response content against specified MIB)
C2F-2	Simple Set	BER / SNMP	SNMP Client	Verify Pre-Set Values. Prepare MIB-verified test SET-Request data. Prepare BER MIB Instance Templates (valid byte encoded responses)	1. Do SET-Request using prepared BER encoded request (verifies protocol and end-point) 2. Verify SET-Response against BER encoded MIB Instance. 3. Inspect that value is SET by doing a GET-Request on the object (see Test Pattern ID 1)
C2F-3	Multiple Get-Set Dialog	BER / SNMP	SNMP Client	Verify Pre-Set Values. Prepare MIB-verified test GET-Request and SET-Request data. Prepare BER MIB Instance Templates (valid byte encoded responses)	A known sequence of GET and SET actions will make up one of these complex dialog patterns. As such each needs to be "reduced" to individual GET – SET tests following Test Patterns 1 and 2 above.
C2F-4	Dynamic Object Dialogs	OER / SNMP & BER / SNMP	SNMP Client	Verify Pre-Set Values. Prepare MIB-verified test GET-Request and SET-Request data. Prepare BER MIB Instance Templates (valid byte encoded responses)	**NOTE: Currently ONLY applies to ASC.** Subscription Setting up dynamic objects involves a short sequence of GET – SET actions using SNMP / BER. Publication Wait for incoming response (update)

					Test will be similar to a GET-Response and should follow the GET-Request Pattern Test (ID #1) above. Updates are OER encoded and test should account for this (e.g., byte encoded MIB Instance template should reflect OER. <u>Other Verification</u> Test of Re-sync capability Test stopping subscription Test re-start of all subscriptions Test of device to handle multiple subscriptions

2.4.1.4 Other Verification

Other verification at the information level include testing for “bad” data in request and response messages. This may include: value out of range, no such object defined, etc.

2.4.2 Application Level Test Procedures

Testing of SNMP BER can be achieved with commonly available SNMP Clients. Most manufacturers provide Ethernet connectivity. Some SNMP-Client manufacturers, however, now support PMPP connectivity.

Testing at this level would include as a minimum:

- Test of an SNMP connection
- Test against a few known objects using SET and GET.

2.4.3 Transport Level Test Procedures (TCP/IP)

Testing of TCP and UDP can be accomplished using the TTCP (Test TCP) Utility, an open source utility originally developed by the military and used by large network equipment vendors, such as Cisco Systems. As TTCP is open source and relatively small, its use as a test tool can be made known to vendors in the specifications.

TTCP is a client/server combination that simply moves TCP packets and is used to test TCP and network bandwidth capability. TCP allows the size of the TCP packet and payload to be specified.

It is expected that a vendor would port TTCP to a device easily or be able to suggest an alternative.

IP testing consists of testing the send and receipt of ICMP packets (mandated in the ITS standards). The PING utility is used widely to test IP connectivity.

2.4.4 Sub Network and Plant Level Test Procedures

EtherNet testing can be accomplilshed using commonly available Ethernet Sniffer Software.

The same holds true for SLIP and PPP – commonly available software, dial-up or direct connect.

PMPP is a customized flavor of HDLC and may required special handling. CHART II’s PMPP implementation is documented. Implementation and testing, therefore, of PMPP may be project-specific.

3 Test Procedures and Tools for Center-to-Center Communications

3.1 Customize and Verify Dialogs and Message Content

The procedures defined herein assume that dialogs for center-to-center communications have been created and are based on project requirements. Moreover, that a project-specific customized WSDL and XML Schema file have been created.

The RTM ties a solution (dialogs, messages and data elements) to the project requirements. For center communications the dialog worksheet that can be extended to facilitate test procedure development.

3.2 Develop and Verify Test Data

It is recommended that a customized projects-specific XML Schema be created that provides a computer-readable form of the message, data frame, and data element definitions. A customized project-specific WSDL file provides computer-readable form of the dialogs (message sequence), transport (HTTP or FTP), encoding rules (XML or SOAP), and end-points (URLs or URIs) for all communications. The XML Schema message, data frame, and data element definitions are just that – definitions – they are note what is sent across a wire. XML Schema definitions for example define value ranges and value formats, but are not the values themselves. Therefore, test data must be created in order to test that an interface has been properly implemented. It is recommended that the test data be written as XML formatted messages, a.k.a. an XML Schema Instance, and that the message be validate against the National and Project-specific XML Schema. In this way the test data (comprised of XML messages) can be proved to conform with the National and Project-specific XML Schema.

3.3 Center-to-Center Communications Test Tools

The table that follows contains a summary of test tools and capabilities as relates to center-to-center communications. The table is organized to show tools as apply to the ITS Framework Communications levels.

NOTE: The information in the table below is based on actual use and evaluation of the test tools. The test tools are commonly available COTS.

Table 3-1. Role of Test Tool as Applies to ITS Center-to-Center Communications

	Information Level (TMDD, IEEE 1512, SAE-J2354, TCIP)				Application Level (NTCIP 2306, IETF)			Transport Level (IETF)		SubNetwork & Plant Level
Tool	**WSDL Verification**	**XML Schema Verification**	**Dialog Pattern Verification**	**Content Verification against XML Schema**	**Content Encoding Verification (SOAP, XML)**	**HTTP**	**FTP**	**TCP**	**IP**	**Project Specific**
XMLSpy 2006	X	X		X	X					
XSV		X		X	X					
SOAP Client			X R-R & Sub-Pub							
SOAP Server			X R-R & Sub-Pub							
HTTP Client			X R-R & One-way			X				
FTP Client			X One-way				X	X	X	
TTCP								X	X	
TraceRoute								X	X	
PING									X	

R-R = Request Response
Sub-Pub = Subscription-Publication

3.4 Center-to-Center Test Procedure Guidance

3.4.1 Information Level Test Procedures

3.4.1.1 Verify the National WSDL and XML Schemas

It is important to ensure that all National WSDL and XML Schemas referenced by the project are available and verified both in their own right and that they are verified to work together – in other words, there are no unreferenced or unresolved elements. This is not a task that has been handled by the National Standards as it is not possible to pre-determine all combinations that might be implemented in projects.

Importantly, the WSDL and XML Schemas must be verified with a tool that will be specified prior to testing.

3.4.1.2 Verify the Project WSDL and XML Schema

In this activity the customized WSDL and XML Schemas are verified, much as outlined in section 3.4.11.

NOTE: There is likely no way to verify that a subset WSDL or XML Schema (the project's) is valid against the National WSDL or XML Schema. Therefore, to prove conformance all information level content must be validated against both the project and national WSDL and XML Schemas.

3.4.1.3 Verification of Dialogs

The following assumptions are pre-conditions to information level dialog testing:

- The project-specific PICS, RTM, WSDL and XML Schema include all mandatory elements of the national standard, and therefore these project-specific documents will ensure conformance with the national standards.
- Test data have been created and verified.
- The dialog is contained in the RTM and Dialog Worksheet.

Dialogs are implemented (dialog patterns) for center-to-center communications as follows:

- One-way = FTP or HTTP GET
- Request-Response = HTTP GET
- Subscription-Publication = HTTP POST

In addition to the general procedures shown in the table before, it is useful to include columns for expected and actual results (these are not shown and would be dialog specific).

Table 3-2. Center-to-Center Communications Dialog Test Procedures (Generalized)

Test Pattern ID	Dialog Pattern	Encoding /Protocol	Tools	Pre-Condition	Procedure
C2C-1	One-way	XML FTP	FTP Client	1. Verify project and WSDL and XML schema are correct. 2. Verify that dialog to be tested is contained in the project WSDL.	1. Do FTP Get (verifies protocol and URL) 2. Save XML File 3. Verify Saved File is XML (encoding is XML) 4. Verify XML File against Project XML Schema (Verifies message content)
C2C-2	One-way	XML HTTP	HTTP Client (e.g., Web Browser)	1. Verify project and WSDL and XML schema are correct. 2. Verify that dialog to be tested is contained in the project WSDL.	1. Do HTTP Get (verifies protocol and URL in WSDL) 2. Save XML File 3. Verify Saved File is XML (encoding is XML) 4. Verify XML File against Project XML Schema (Verifies message content)
C2C-3	Request – Response	XML HTTP	HTTP Client (e.g., Web Browser)	1. Verify project and WSDL and XML schema are correct. 2. Verify that dialog to be tested is contained in the project WSDL.	1. Do HTTP POST (verifies protocol and URL in WSDL) 2. Save XML File 3. Verify Saved File is XML (encoding is XML) 4. Verify XML File against Project XML Schema (Verifies message content)
C2C-4	Request – Response	SOAP HTTP	SOAP HTTP Client	1. Verify project and WSDL and XML schema are correct. 2. Verify that dialog to be tested is contained in the project WSDL. 3. Verify that the XML Request Message is valid against project schema.	1. Send XML Request Message 2. Receive XML Response Message 3. Save XML Response Message to a File 4. Verify Saved File is SOAP XML (encoding is SOAP) 5. Verify SOAP XML File against Project XML Schema (Verifies message content)
C2C-5	Subscription – Publication	SOAP HTTP	SOAP HTTP Client for Subscription SOAP HTTP Server (listener) to Publication	Set up Listener (SOAP Server)	<u>Subscription</u> 1. Send XML Request Message (Subscription) 2. Receive XML Response Message (Subscription Receipt) 3. Save XML Response Message to a File 4. Verify Saved File is SOAP XML (encoding is SOAP) 5. Verify SOAP XML File against Project XML Schema (Verifies message content)

					Publication Wait for incoming response (update message) 1. Receive XML Response Message (Publication) 2. Allow Listener to send Publication Receipt to Publisher 3. Save XML Response Message to a File 4. Verify Saved File is SOAP XML (encoding is SOAP) 5. Verify SOAP XML File against Project XML Schema (Verifies message content) 6. Verify that Counter is Incrementing Properly 7. Repeat 1 -6 one more time Other Verification Test of Re-sync capability Test stopping subscription Test re-start of all subscriptions Test of server to handle multiple subscriptions

3.4.1.4 Other Verification

Other verification at the information level include testing for “bad” data in request and response messages. This may include: value out of range, no such message/data element defined, etc.

3.4.2 Application Level Test Procedures

HTTP GET can be tested using commonly available HTTP Client software.

Test of FTP GET – use commonly available FTP Client software.

Test of HTTP POST – use commonly available HTTP Client software and servers.

SOAP encoding requires that the SOAP Client test utility provide for document-literal and NOT rpc style formatting – this is clearly defined for NTCIP 2306 compliant WSDL. With that caveat, SOAP Clients and Servers are commonly available.

3.4.3 Transport Level Test Procedures (TCP/IP)

See discussion in Section 2.4.3. The same applies.

3.4.4 Sub Network and Plant Level Test Procedures

See discussion in Section 2.4.4. With the exception of the discussion about PMPP, the same applies.