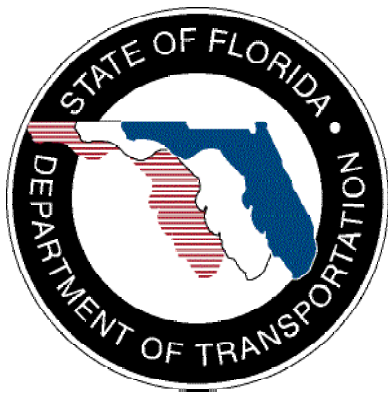


Statewide Transportation Management Center Software Library System:

Transportation Sensor System Subsystem Interface Control Document

STMCSLS-TSS-SUB-ICD-1.0.0-Design



Prepared for:

Florida Department of Transportation
ITS Office
605 Suwannee Street, M.S. 90
Tallahassee, Florida 32399-0450
(850) 410-5600

November 21, 2003

Document Control Panel			
File Name:	STMCSLS-TSS-SUB-ICD-1.0.0-Design.doc		
File Location:	STMCSLS CM Repository		
CDRL:	2-7.1		
	Name	Initial	Date
Created By:	Lynne Randolph, SwRI	LAR	11/18/03
Reviewed By:	Steve Dellenback, SwRI	SWD	11/19/03
	Steve Novosad, SwRI	SEN	11/21/03
Modified By:			
Completed By:			

Table of Contents

1. Scope	1
1.1 Document Identification	1
1.2 Project Overview	1
1.3 Related Documents	2
1.4 Contacts	3
2. Data	4
2.1 Message Format	4
2.2 Documentation Key	5
2.3 XML Transactions	5
2.3.1 Request.....	6
2.3.2 Response	8
2.4 Initial Client Communication	13
2.4.1 Authenticate	13
2.4.2 Subscribe.....	16
2.4.3 Other commands	16
2.5 Example XML Commands	17
3. Interface	18
3.1 Network Connection	18
3.2 Command Sequence	18
3.2.1 Connection Establishment	18
3.2.2 Client Data Requests.....	18
3.2.3 Periodic Server Updates.....	18
4. Notes	19

List of Figures

Figure 1.1 – High-Level Architectural Concept	2
Figure 2.1 – Illustration of Byte Ordering	4
Figure 2.2 – TransactionType Abstract Type	5
Figure 2.3 – Required refId Element	6
Figure 2.4 – Required icdVersion Element.....	6
Figure 2.5 – Optional debug Element	6
Figure 2.6 – RequestType Abstract Type	7
Figure 2.7 – Optional username Element	8
Figure 2.8 – Optional securityToken Element.....	8
Figure 2.9 – ResponseType Abstract Type.....	10
Figure 2.10 – Required error Element	11
Figure 2.11 – Required errorCode Element.....	11
Figure 2.12 – Optional errorMap Element.....	11
Figure 2.13 – Required errorString Element.....	12
Figure 2.14 – Optional errorData Element	12
Figure 2.15 – Optional equipmentId Element.....	12
Figure 2.16 – Required data Element	13
Figure 2.17 – authenticateReq	14
Figure 2.18 – username.....	14
Figure 2.19 – password.....	15
Figure 2.20 – authenticateResp.....	15
Figure 2.21 – authenticateData	16

List of Acronyms

ATMS	Advanced Traffic Management System
DOT	Department of Transportation
FDOT	Florida Department of Transportation
IM.....	Incident Management
ITS.....	Intelligent Transportation Systems
ITN.....	Invitation to Negotiate
STMCSLS.....	Statewide Transportation Management Center Software Library System
SwRI	Southwest Research Institute
TMC	Traffic Management Center
TSS.....	Transportation Sensor Systems
XML.....	eXtensible Markup Language

REVISION HISTORY

Revision	Date	Changes
1.0.0-Design	November 21, 2003	Initial Release

1. Scope

1.1 Document Identification

This Interface Control Document (ICD) describes the interface between the Transportation Sensor Subsystem (TSS) and other Statewide Transportation Management Center Software Library System (STMCSLS) subsystems. This ICD defines Extensible Markup Language (XML) schemas upon which XML requests shall be based upon in communicating to the TSS Subsystem.

1.2 Project Overview

The Florida Department of Transportation (FDOT) is conducting a program that is developing a STMCSLS. The STMCSLS is a set of Intelligent Transportation System (ITS) software that allows the control of roadway devices as well as information exchange across a variety of transportation agencies. The goal of the STMCSLS is to have a common software base that can be deployed throughout the state of Florida. The STMCSLS development effort is based on ITS software available from both the states of Texas and Maryland; significant customization of the software is being performed as well as the development of new software modules. The following figure provides a graphical view of the software to be developed:

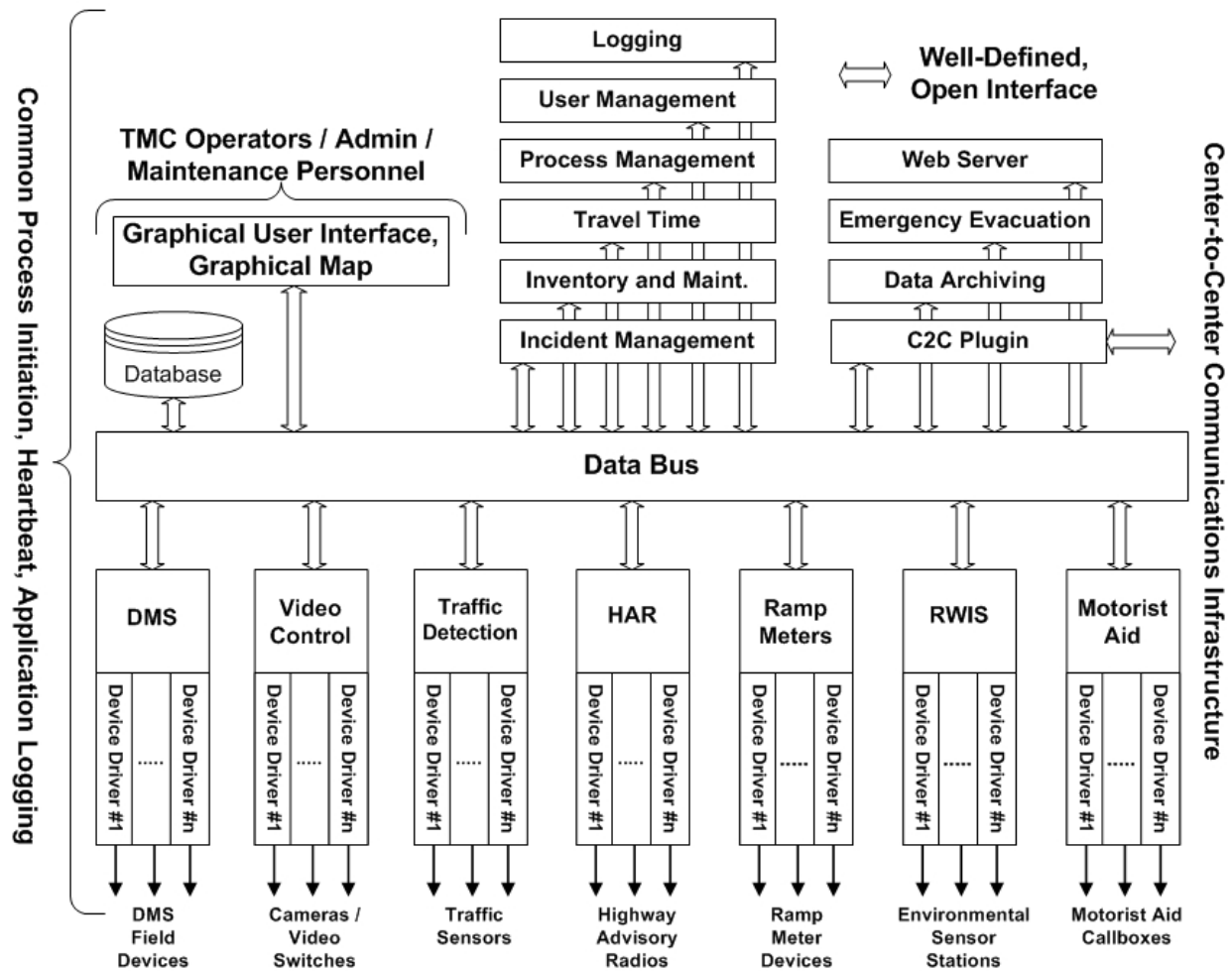


Figure 1.1 – High-Level Architectural Concept

The STMCSLS development effort spans approximately two years. After the development, the software will be deployed to a number of cities throughout Florida and support activities will be performed.

1.3 Related Documents

The following documents were used to develop this document:

- SwRI® Qualification Response: *Response to the Invitation to Negotiate (ITN): Statewide Transportation Management Center Software Library System, Negotiation Number: ITN-DOT-02/03-9025-RR*, SwRI Proposal No. 10-35924, dated: November 18, 2002.
- SwRI Technical Proposal: *Technical Proposal for Invitation to Negotiate (ITN): Statewide Transportation Management Center Software Library System, Negotiation Number: ITN-DOT-02/03-9025-RR*, SwRI Proposal No. 10-35924, dated: January 31, 2003.

- SwRI Cost Proposal: *Cost Proposal for Invitation to Negotiate (ITN): Statewide Transportation Management Center Software Library System, Negotiation Number: ITN-DOT-02/03-9025-RR*, SwRI Proposal No. 10-35924, dated: January 31, 2003.
- SwRI BAFO letter: *Southwest Research Institute[®] Proposal No. 10-35924, “Invitation to Negotiate (ITN): Statewide Transportation Management Center Software Library System”*, Reference: Negotiation Number: ITN-DOT-02/03-9025-RR, dated: May 5, 2003.
- FDOT procurement document: *Invitation To Negotiate (ITN), Negotiation Number: ITN-DOT-02/03-9025-RR, Statewide Transportation Management Center Software Library System*, dated: October 21, 2002.
- FDOT Scope of Services: *Statewide Transportation Management Center Software Library System: Scope of Services*, September 22, 2003.
- FDOT Requirements Document: *Statewide Transportation Management Center Software Library System: Requirements Specification*, June 3, 2003.
- World Wide Web Consortium (W3) website: <http://www.w3.org>.

1.4 Contacts

The following are contact persons for the STMCSLS project:

- Chester Chandler, ITS Central Office, chester.chandler@dot.state.fl.us, 850-410-5600
- Liang Hsia, FDOT Project Manager, liang.hsia@dot.state.fl.us, 850-410-5615
- John Bonds, Senior ITS Specialist, jbonds@pbsj.com, 408-873-2514
- David Chang, ITS Specialist, David.Chang@dot.state.fl.us, 850-410-5622
- Steve Dellenback, SwRI Project Manager, sdellenback@swri.org, 210-522-3914
- Robert Heller, SwRI Software Project Manager, rheller@swri.org, 210-522-3824
- Charlie Wallace, PBF Deputy Project Manager, WallaceC@pbworld.com, 352-374-6635
- John Schumitz, PBF Software Project Manager, schumitz@pbworld.com, 301-816-1852

The following are contacts that will be used by the STMCSLS project team to assure consistency with other FDOT projects and FDOT procedures:

- Dan Baxter, PB Farradyne, FDOT C2C Project, baxter@pbworld.com, 407-587-7809
- David Lambert, University of North Florida, RWIS, jlambert@unf.edu, 904-620-3881
- Bob Colins, PBS&J, Emergency Evacuation, bobcolins@pbsj.com, 850-575-1800
- John Fain, FDOT, Comptroller, john.fain@dot.state.fl.us, 850-921-7332
- Jerry Bloodgood, McCain, Ramp Metering
- Leslie Jacobson, PB Farradyne, Ramp Metering, jacobsonl@pbworld.com, 206-382-5290

2. Data

The following sections detail the transactions that can be exchanged between the TSS Subsystem and a client application. Transactions begin with the size of the data being sent, followed by the compression type and the XML command being sent.

Transactions are sent from the application (client) to the TSS Subsystem (server) and from the TSS Subsystem to the application. The data formats expected are identical for requests and responses. The actual XML being sent determines the request or response that is being transmitted.

2.1 Message Format

The data items for each data type are described in detail, including the data type and size, and a detailed description (if specific values apply to the item). All integer and bitmap data will be in big endian byte order (i.e. the least significant byte is the farthest to the right). Bits are labeled from right (#0) to left of a byte or word. Strings are not null terminated and are of variable length.

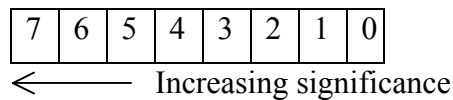


Figure 2.1 – Illustration of Byte Ordering

Table 2.1 – Transaction Parameters

Data Item Description	Data Type and Size	Detailed Data Description
Message Size	Integer – 4 bytes	The size of the XML being transmitted (compressed or uncompressed).
Original Message Size	Integer – 4 bytes	This value is used only if compression is being used. This is the size of the original message before compression. ISO-8859-1 is used compress/decompress. If original message size is equal to message size then compression is not used. Note: ZLIB is the only supported compression algorithm. More information on ZLIB is available at: http://www.gzip.org/zlib/ .
Adler-32 checksum	Integer – 8 bytes	The Adler-32 checksum for the compressed data, 0 if not compressed.
XML	String - Varied	Variable, see schema information for specific requests and responses.

XML schemas, sample XML and documentation for the requests and responses can be viewed on the [schema specification page](#).

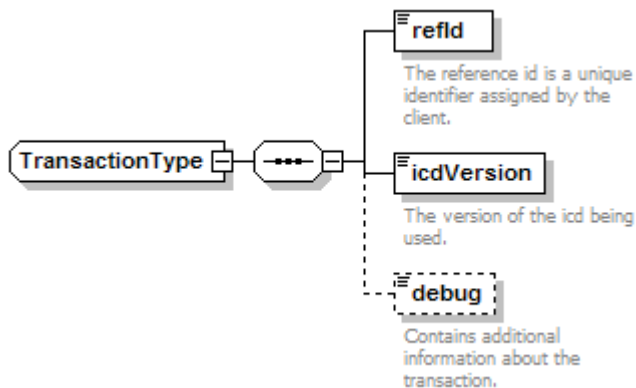
2.2 Documentation Key

The documentation for the XML schemas uses notation that is unique to this type of document. The figures in the [Schema Key](#) show the meanings for various symbols used.

2.3 XML Transactions

The diagram in Figure 2.2 shows the transaction type used by all requests and responses sent to or from the Interface. The transaction type is an abstract type that each request or response must extend. The *refId* is an identifier assigned by the client to uniquely identify this transaction. Responses generated by the Interface will return the *refId* that was designated in the request. The *icdVersion* will verify that the client is using the appropriate ICD and XML schemas.

diagram



children [refId](#) [icdVersion](#) [debug](#)

used by complexTypes [RequestType](#) [ResponseType](#)

```

source <xs:complexType name="TransactionType" abstract="true">
  <xs:sequence>
    <xs:element name="refId" type="xs:string">
      <xs:annotation>
        <xs:documentation>The reference id is a unique identifier assigned by the client.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="icdVersion" type="xs:string">
      <xs:annotation>
        <xs:documentation>The version of the icd being used.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="debug" type="xs:string" minOccurs="0">
      <xs:annotation>
        <xs:documentation>Contains additional information about the transaction.</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
  
```

Figure 2.2 – TransactionType Abstract Type

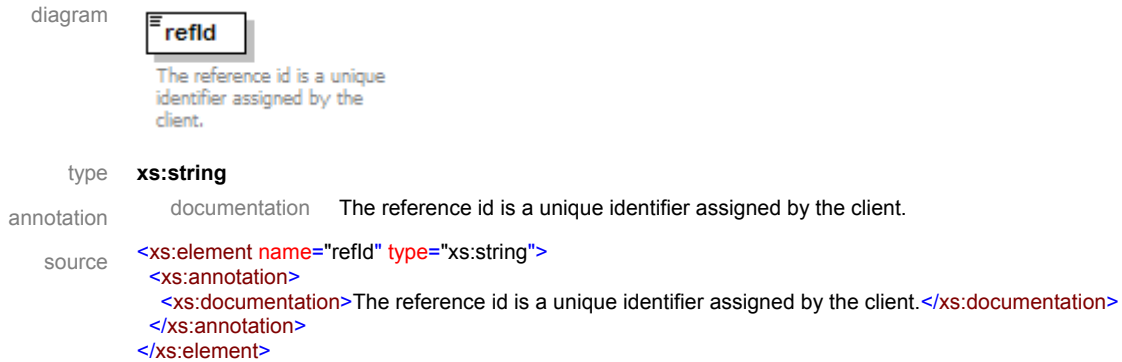


Figure 2.3 – Required refId Element

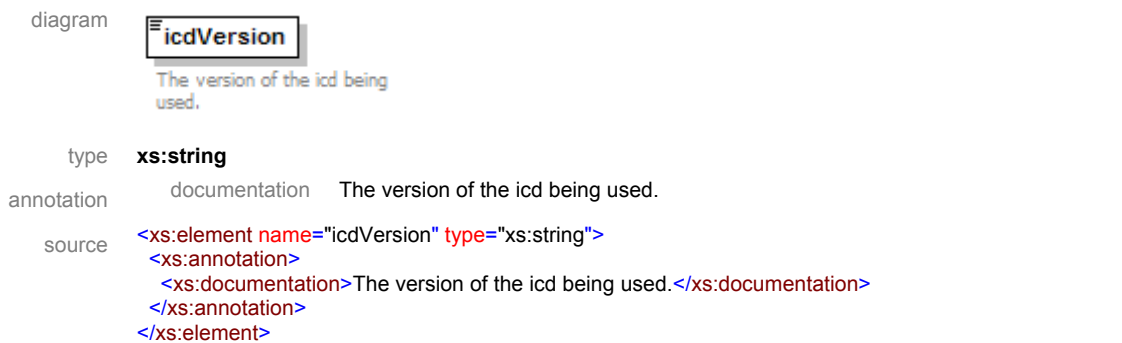


Figure 2.4 – Required icdVersion Element

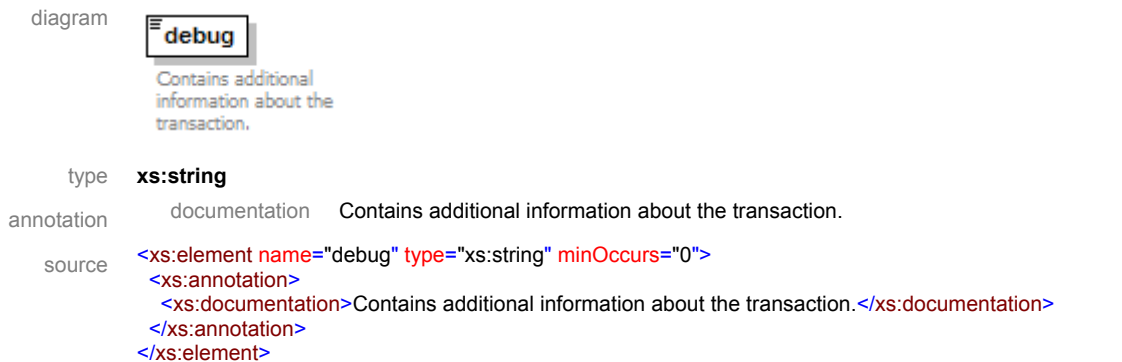
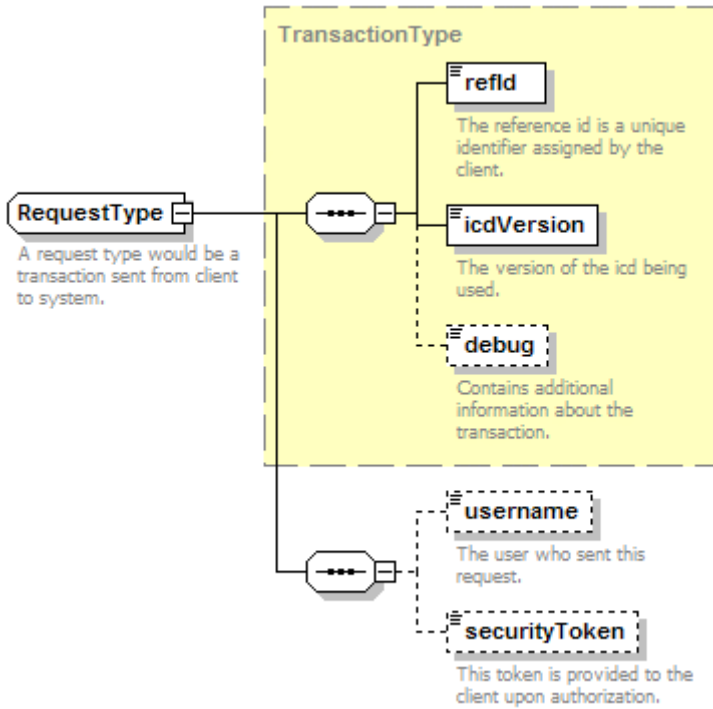


Figure 2.5 – Optional debug Element

2.3.1 Request

A request is a transaction that contains two additional data fields: *username* and *securityToken*. As shown in Figure 2.7, the *username* is a string representing the client who has been authenticated. The *securityToken* is a string that is returned to a client upon authentication to a provider subsystem. The documentation can be viewed on the [Schema web page](#).

diagram



type	extension of TransactionType					
children	refId icdVersion debug username securityToken					
attributes	Name	Type	Use	Default	Fixed	Annotation
	resourceType	identifier	optional			
annotation	documentation	A request type would be a transaction sent from client to system.				
source	<pre> <xs:complexType name="RequestType" abstract="true"> <xs:annotation> <xs:documentation>A request type would be a transaction sent from client to system.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="TransactionType"> <xs:sequence> <xs:element name="username" type="identifier" minOccurs="0"> <xs:annotation> <xs:documentation>The user who sent this request.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="securityToken" minOccurs="0"> <xs:annotation> <xs:documentation>This token is provided to the client upon authorization.</xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:string"> <xs:minLength value="6"/> <xs:maxLength value="30"/> </xs:restriction> </xs:simpleType> </xs:element> </xs:sequence> <xs:attribute name="resourceType" type="identifier" use="optional"/> </xs:extension> </xs:complexContent> </xs:complexType> </pre>					

Figure 2.6 – RequestType Abstract Type

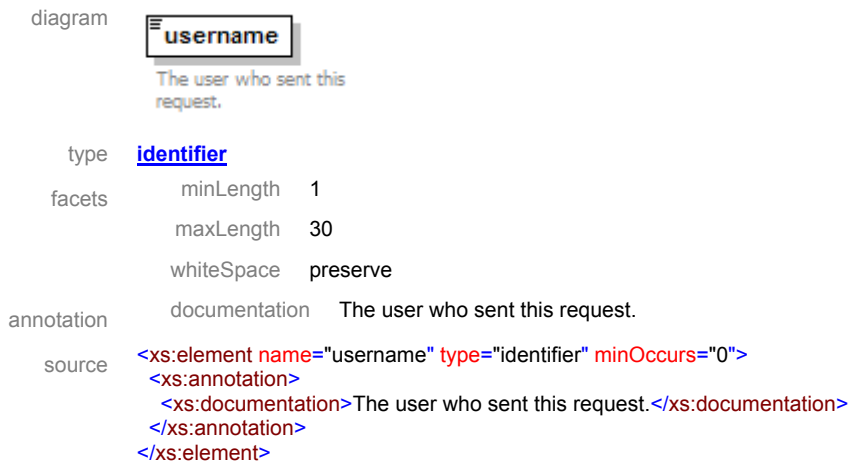


Figure 2.7 – Optional username Element

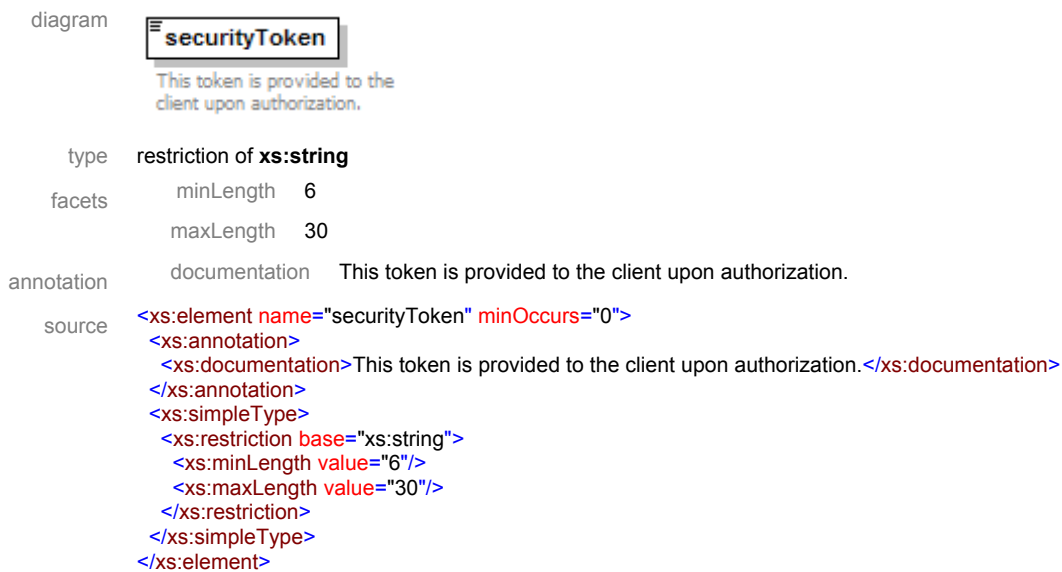
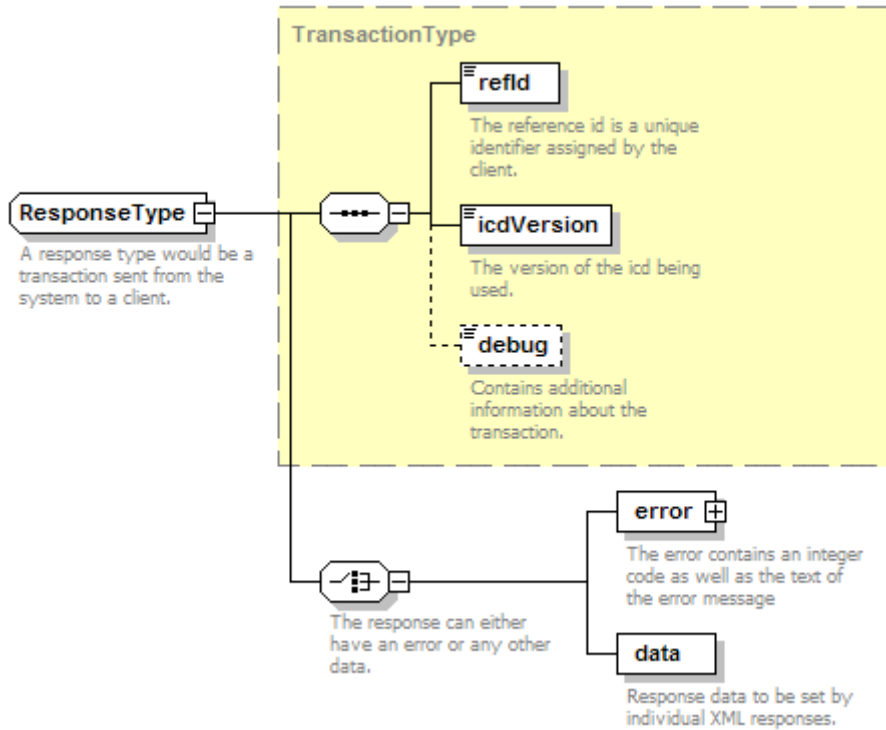


Figure 2.8 – Optional securityToken Element

2.3.2 Response

The response is a transaction that contains either an error or the response data. An error returned by the Interface will contain an error code, which is an integer value, and an error string containing the text of the error message. If there is no error, the response data is returned. For responses where the client has added or modified data, the response will include the new data. The data is returned in the response to allow clients who have subscribed to the data changes to receive the same response and update their data. The documentation can be viewed on the [Schema web page](#).

diagram



type extension of [TransactionType](#)

children [refid](#) [icdVersion](#) [debug](#) [error](#) [data](#)

annotation documentation A response type would be a transaction sent from the system to a client.

source

```

<xs:complexType name="ResponseType" abstract="true">
  <xs:annotation>
    <xs:documentation>A response type would be a transaction sent from the system to a client.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="TransactionType">
      <xs:choice>
        <xs:annotation>
          <xs:documentation>The response can either have an error or any other data.</xs:documentation>
        </xs:annotation>
        <xs:element name="error">
          <xs:annotation>
            <xs:documentation>The error contains an integer code as well as the text of the error message</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element name="errorCode" type="xs:integer">
                <xs:annotation>
                  <xs:documentation>The error code may be used by the client.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="errorMap" type="xs:string" minOccurs="0">
                <xs:annotation>
                  <xs:documentation>Name of table to lookup the error code.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="errorString" type="xs:string">
                <xs:annotation>
                  <xs:documentation>This string contains the error text set by the originating process.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="errorData" type="xs:string" minOccurs="0">
                <xs:annotation>
  
```

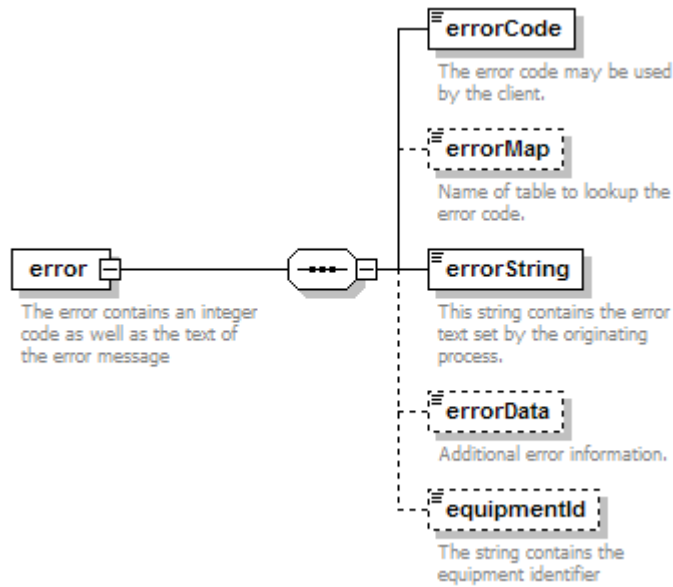
```

        <xs:documentation>Additional error information.</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="equipmentId" type="xs:string" minOccurs="0">
    <xs:annotation>
        <xs:documentation>The string contains the equipment identifier</xs:documentation>
    </xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="data" type="responseData">
    <xs:annotation>
        <xs:documentation>Response data to be set by individual XML responses.</xs:documentation>
    </xs:annotation>
</xs:element>
</xs:choice>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

Figure 2.9 – ResponseType Abstract Type

diagram



children [errorCode](#) [errorMap](#) [errorString](#) [errorData](#) [equipmentId](#)

annotation documentation The error contains an integer code as well as the text of the error message

```

source <xs:element name="error">
  <xs:annotation>
    <xs:documentation>The error contains an integer code as well as the text of the error message</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="errorCode" type="xs:integer">
        <xs:annotation>
          <xs:documentation>The error code may be used by the client.</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="errorMap" type="xs:string" minOccurs="0">
        <xs:annotation>
          <xs:documentation>Name of table to lookup the error code.</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>

```

```

<xs:element name="errorString" type="xs:string">
  <xs:annotation>
    <xs:documentation>This string contains the error text set by the originating process.</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="errorData" type="xs:string" minOccurs="0">
  <xs:annotation>
    <xs:documentation>Additional error information.</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="equipmentId" type="xs:string" minOccurs="0">
  <xs:annotation>
    <xs:documentation>The string contains the equipment identifier</xs:documentation>
  </xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>

```

Figure 2.10 – Required error Element

diagram  The error code may be used by the client.

type **xs:integer**

annotation documentation The error code may be used by the client.

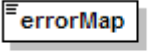
source

```

<xs:element name="errorCode" type="xs:integer">
  <xs:annotation>
    <xs:documentation>The error code may be used by the client.</xs:documentation>
  </xs:annotation>
</xs:element>

```

Figure 2.11 – Required errorCode Element

diagram  Name of table to lookup the error code.

type **xs:string**

annotation documentation Name of table to lookup the error code.

source

```

<xs:element name="errorMap" type="xs:string" minOccurs="0">
  <xs:annotation>
    <xs:documentation>Name of table to lookup the error code.</xs:documentation>
  </xs:annotation>
</xs:element>

```

Figure 2.12 – Optional errorMap Element

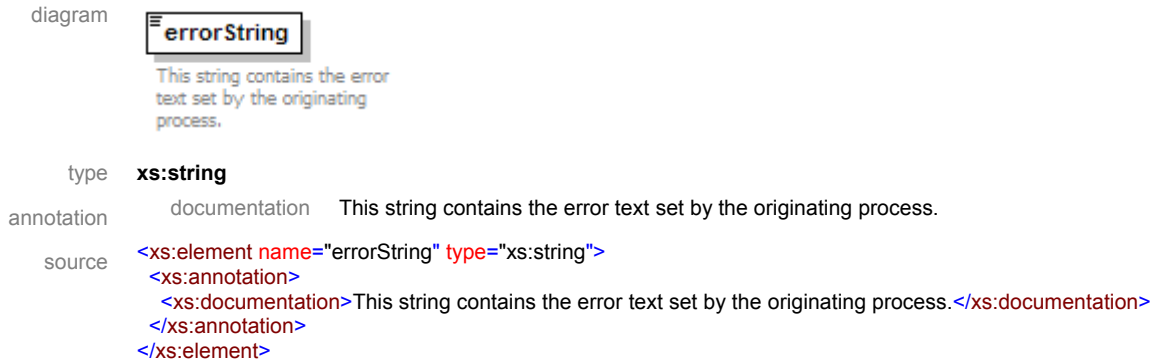


Figure 2.13 – Required errorString Element

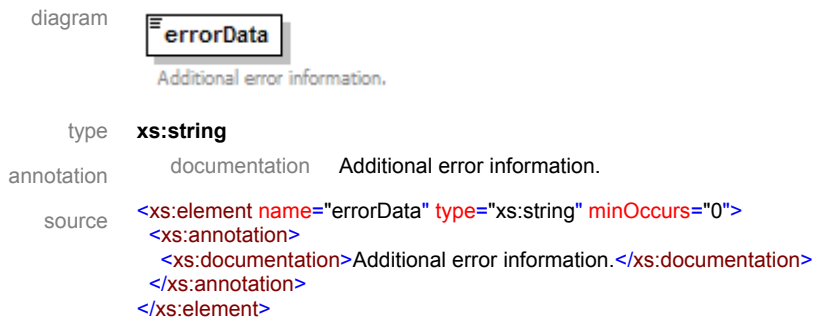


Figure 2.14 – Optional errorData Element

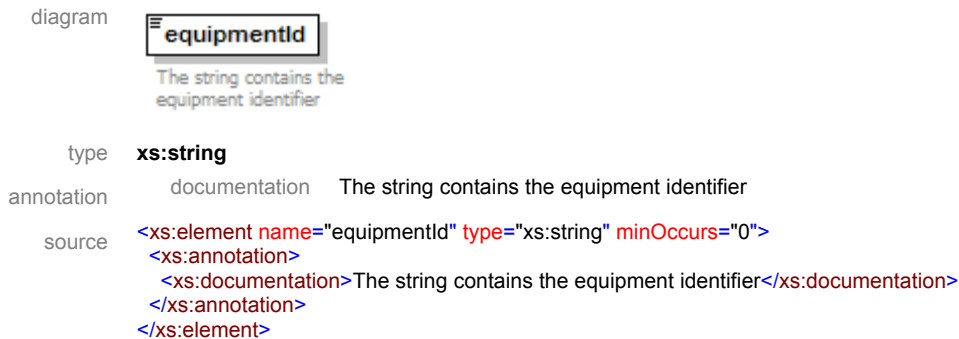


Figure 2.15 – Optional equipmentId Element

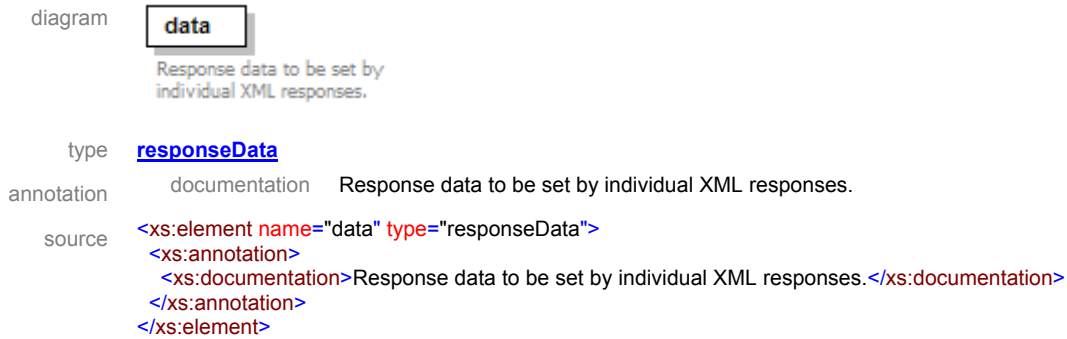


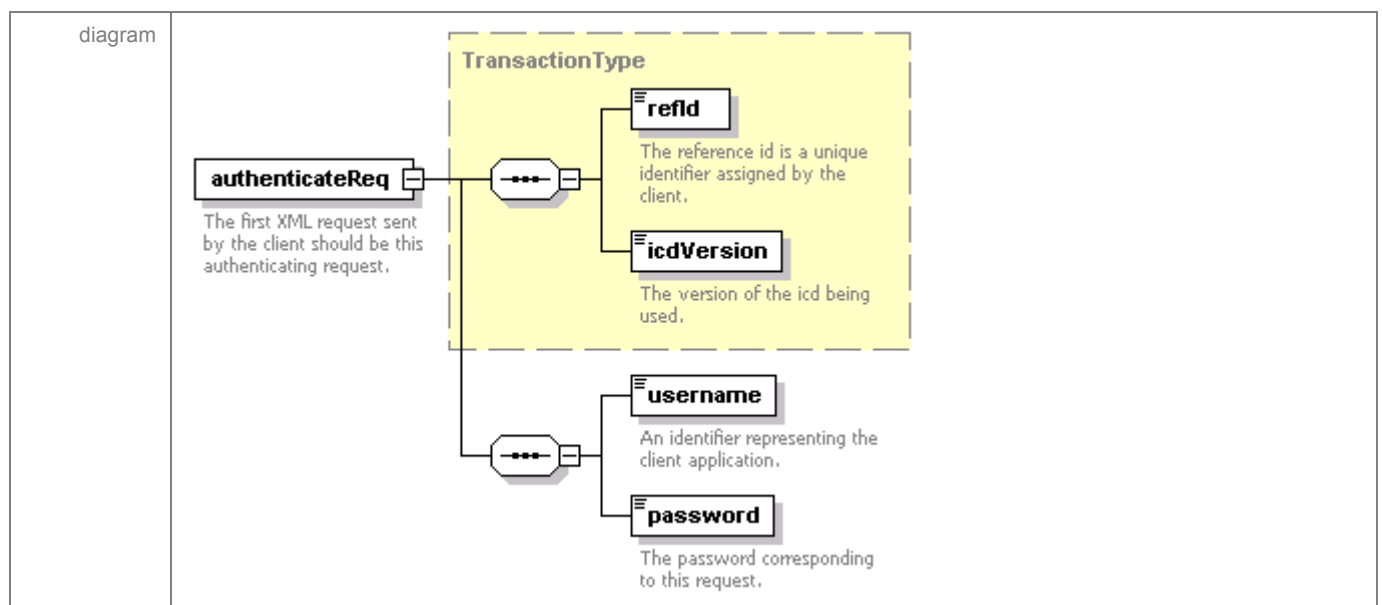
Figure 2.16 – Required data Element

2.4 Initial Client Communication

2.4.1 Authenticate

Before any other commands can be sent, a client must send an authenticateRequest to register with the TSS Subsystem. The authenticate request is a transaction type which contains two additional fields: *username* and *password*. The *username* is the name of the application or user that is connecting to the Interface. The application/user must have an associated password that the Interface can retrieve from the database. The *password* sent as part of this request will be encrypted using Message Digest 5 (MD5) hashing.

The system uses the *username* and *password* to verify the client’s privileges. If the authentication is successful, a *securityToken* will be returned to the client. If not successful, an error message will be returned. The *securityToken* returned by the Interface to the client must be sent with each additional request and will be used to validate the client’s ability to perform the request.



type	extension of TransactionType
children	refId icdVersion username password
annotation	documentation The first XML request sent by the client should be this authenticating request.
source	<pre> <xs:element name="authenticateReq"> <xs:annotation> <xs:documentation>The first XML request sent by the client should be this authenticating request.</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="TransactionType"> <xs:sequence> <xs:element name="username" type="identifier"> <xs:annotation> <xs:documentation>An identifier representing the client application.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="password"> <xs:annotation> <xs:documentation>The password corresponding to this request.</xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:hexBinary"> <xs:maxLength value="30"/> <xs:minLength value="6"/> </xs:restriction> </xs:simpleType> </xs:element> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </xs:element> </pre>

Figure 2.17 – authenticateReq

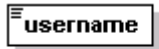
diagram	 <p>An identifier representing the client application.</p>
type	identifier
facets	minLength 1 maxLength 30 whiteSpace preserve
annotation	documentation An identifier representing the client application.
source	<pre> <xs:element name="username" type="identifier"> <xs:annotation> <xs:documentation>An identifier representing the client application.</xs:documentation> </xs:annotation> </xs:element> </pre>

Figure 2.18 – username

diagram	 <p>The password corresponding to this request.</p>
---------	--

type	restriction of xs:hexBinary
facets	minLength 6 maxLength 30
annotation	documentation The password corresponding to this request.
source	<pre> <xs:element name="password"> <xs:annotation> <xs:documentation>The password corresponding to this request.</xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:hexBinary"> <xs:maxLength value="30"/> <xs:minLength value="6"/> </xs:restriction> </xs:simpleType> </xs:element> </pre>

Figure 2.19 – password

The response for an authenticate request includes the security token. The security token is then sent with each additional request from the client.

diagram	
type	extension of ResponseType
children	refId icdVersion error data
annotation	documentation Response received for an authenticate request
source	<pre> <xs:element name="authenticateResp"> <xs:annotation> <xs:documentation>Response received for an authenticate request</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="ResponseType"/> </xs:complexContent> </xs:complexType> </xs:element> </pre>

Figure 2.20 – authenticateResp

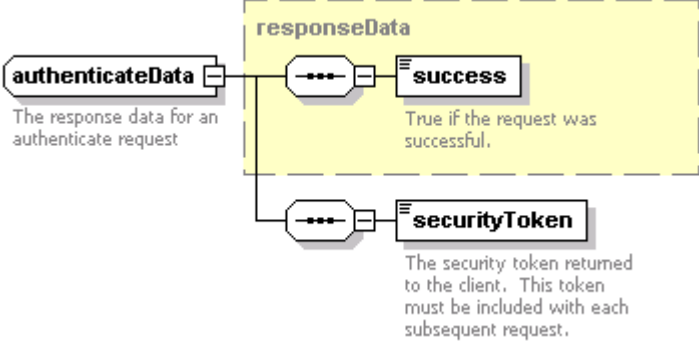
<p>diagram</p>	
<p>type</p>	<p>extension of responseData</p>
<p>children</p>	<p>success securityToken</p>
<p>annotation</p>	<p>documentation The response data for an authenticate request</p>
<p>source</p>	<pre> <xs:complexType name="authenticateData" abstract="0"> <xs:annotation> <xs:documentation>The response data for an authenticate request</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="responseData"> <xs:sequence> <xs:element name="securityToken"> <xs:annotation> <xs:documentation>The security token returned to the client. This token must be included with each subsequent request.</xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:string"> <xs:minLength value="6"/> <xs:maxLength value="30"/> </xs:restriction> </xs:simpleType> </xs:element> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Figure 2.21 – authenticateData

2.4.2 Subscribe

Another request that a client might send to the system is a subscribe request. The client can specify what types of data updates should be sent. Then, if that data changes at a later time, the client will receive unsolicited responses with updated data. The subscribe response will return true values for data to which the client has successfully subscribed. False values will be returned for data types to which the subscription failed. The subscribe may fail if the client does not have permission to retrieve the data types requested. The documentation can be viewed on the [Schema web page](#).

2.4.3 Other commands

Other commands schemas and documentation can be viewed on the [Schema web page](#).

2.5 Example XML Commands

Sample XML files for typical requests from clients and the associated responses may be accessed on the [Schema web page](#).

3. Interface

The following sections describe the interface between the TSS Subsystem and client applications.

3.1 Network Connection

A TCP/IP connection will be utilized to exchange data. The TSS Subsystem will act as the server and applications will act as clients. A session begins when the client connects to the server and sends the authenticate request. The session ends when the connection closes. If the connection between the client and server is severed for any reason, the client must start anew by opening a TCP connection and sending the authenticate request.

3.2 Command Sequence

The following sections describe the elements of a session between a client and the server. The session begins with the opening of a connection and the client sending an authenticate request. Then the client may request data or an action by the TSS Subsystem. The system may also notify the client of raw and smoothed traffic data or alarms as periodic updates. The session ends when the connection is closed.

3.2.1 Connection Establishment

When the client connects to the server, the authenticate request must be sent. No other commands will be accepted before this. The authenticate command provides the server with information to allow the server to determine the client's privileges; what commands the client can use and what data are available to the client. The security token returned from the authenticate request will be associated with the client and used to determine the privileges for each successive request.

3.2.2 Client Data Requests

Once the authenticate response has been received without error, the client can initiate any other requests. The format of these requests can be viewed by opening the appropriate schema or sample XML.

3.2.3 Periodic Server Updates

While the client is connected, the server will send periodic updates (asynchronous updates) to the client. These notifications will only be sent if the client has previously subscribed to receive the type of data being modified. Updates occur in the form of unsolicited responses from the server (i.e. link update or alarm triggered).

4. Notes

Information about XML and schemas can be found at the World Wide Web Consortium (W3) website at <http://www.w3.org>