ATIS-0300002.2018

XML Schema Interface for POTS Service Test

AMERICAN NATIONAL STANDARD FOR TELECOMMUNICATIONS

As a leading technology and solutions development organization, the Alliance for Telecommunications Industry Solutions (ATIS) brings together the top global ICT companies to advance the industry's most pressing business priorities. ATIS' nearly 200 member companies are currently working to address the All-IP transition, 5G, network functions virtualization, big data analytics, cloud services, device solutions, emergency services, M2M, cyber security, network evolution, quality of service, billing support, operations, and much more. These priorities follow a fast-track development lifecycle — from design and innovation through standards, specifications, requirements, business use cases, software toolkits, open source solutions, and interoperability testing.

ATIS is accredited by the American National Standards Institute (ANSI). The organization is the North American Organizational Partner for the 3rd Generation Partnership Project (3GPP), a founding Partner of the oneM2M global initiative, a member of the International Telecommunication Union (ITU), as well as a member of the Inter-American Telecommunication Commission (CITEL). For more information, visit www.atis.org.

AMERICAN NATIONAL STANDARD

Approval of an American National Standard requires review by ANSI that the requirements for due process, consensus, and other criteria for approval have been met by the standards developer.

Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that a concerted effort be made towards their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether he has approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not conforming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give an interpretation of any American National Standard. Moreover, no person shall have the right or authority to issue an interpretation of an American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

CAUTION NOTICE: This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken periodically to reaffirm, revise, or withdraw this standard. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.

Notice of Disclaimer & Limitation of Liability

The information provided in this document is directed solely to professionals who have the appropriate degree of experience to understand and interpret its contents in accordance with generally accepted engineering or other professional standards and applicable regulations. No recommendation as to products or vendors is made or should be implied.

NO REPRESENTATION OR WARRANTY IS MADE THAT THE INFORMATION IS TECHNICALLY ACCURATE OR SUFFICIENT OR CONFORMS TO ANY STATUTE, GOVERNMENTAL RULE OR REGULATION, AND FURTHER, NO REPRESENTATION OR WARRANTY IS MADE OFMERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR AGAINST INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS. ATIS SHALL NOT BE LIABLE, BEYOND THE AMOUNT OF ANY SUM RECEIVED IN PAYMENT BY ATIS FOR THIS DOCUMENT, AND IN NO EVENT SHALL ATIS BE LIABLE FOR LOST PROFITS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES. ATIS EXPRESSLY ADVISES THAT ANY AND ALL USE OF OR RELIANCE UPON THE INFORMATION PROVIDED IN THIS DOCUMENT IS AT THE RISK OF THE USER.

NOTE - The user's attention is called to the possibility that compliance with this standard may require use of an invention covered by patent rights. By publication of this standard, no position is taken with respect to whether use of an invention covered by patent rights will be required, and if any such use is required no position is taken regarding the validity of this claim or any patent rights in connection therewith. Please refer to [http://www.atis.org/legal/patentinfo.asp] to determine if any statement has been filed by a patent holder indicating a willingness to grant a license either without compensation or on reasonable and non-discriminatory terms and conditions to applicants desiring to obtain a license.

ATIS-0300002.2018, *XML Schema Interface for POTS Service Test*

Is an American National Standard developed by the ATIS **Telecom Management and Operations Committee (TMOC)**.

Published by

Alliance for Telecommunications Industry Solutions
1200 G Street, NW, Suite 500
Washington, DC 20005

ATIS-0300002.2018

(Revision of ATIS-0300002.2013)

American National Standard for Telecommunications

XML Schema Interface for POTS Service Test

Alliance for Telecommunications Industry Solutions

Approved June 2018

American National Standards Institute, Inc.

Abstract

This standard provides an XML schema information model for POTS Service Test based on ATIS-0300262 and an XML schema interface for POTS Service Test function specified in the same ANSI standard.

Foreword

The information contained in this Foreword is not part of this American National Standard (ANS) and has not been processed in accordance with ANSI's requirements for an ANS. As such, this Foreword may contain material that has not been subjected to public review or a consensus process. In addition, it does not contain requirements necessary for conformance to the Standard.

The Alliance for Telecommunication Industry Solutions (ATIS) serves the public through improved understanding between carriers, customers, and manufacturers. The Telecom Management and Operations Committee (TMOC) – formerly T1M1 -- develops operations, administration, maintenance and provisioning standards, and other documentation related to Operations Support System (OSS) and Network Element (NE) functions and interfaces for communications networks - with an emphasis on standards development related to U.S.A. communication networks in coordination with the development of international standards.

ANSI guidelines specify two categories of requirements: mandatory and recommendation. The mandatory requirements are designated by the word *shall* and recommendations by the word *should*. Where both a mandatory requirement and a recommendation are specified for the same criterion, the recommendation represents a goal currently identifiable as having distinct compatibility or performance advantages.

Suggestions for improvement of this document are welcome. They should be sent to the Alliance for Telecommunications Industry Solutions, TMOC, 1200 G Street NW, Suite 500, Washington, DC 20005.

At the time it approved this document, TMOC, which is responsible for the development of this Standard, had the following leadership:

P. Galarza, TMOC Chair (iconectiv)
T. Barrett, Technical Editor (AT&T)

Table of Contents

Table of Figures

Table of Tables

American National Standard for Telecommunications –

XML Schema Interface for POTS Service Test

1 Introduction

1.1 Scope

The scope of this document is to develop an American National Standards (ANS) standard for testing services by extending the object model in ITU-T Recommendation X.745 for the purpose of testing Plain Old Telephone Service (POTS). This standard will allow service customers to request a test and to receive subsequent test results for one of their POTS services.

1.2 Application

The model described in this standard defines management information to be exchanged for testing POTS services through use of an uncontrolled test function. The application of this model is for the service level. The service level test functions are documented in ITU-T Recommendation M.3400.

The information exchanged to perform the above test function utilizes the *tML Framework* (ITU-T Recommendation M.3030). ITU-T Recommendation M.3020 defines a three-phase methodology for developing interface specifications related to management information exchanges. The requirements phase is derived from existing standard ITU-T Recommendation X.745 for the generic test management architecture and from previous work done in ATIS-0300262 for specific POTS service level tests including tone tests. The requirements are shown here in clause 3. The second, formal analysis phase from M.3020 is provided in the UML model of clause 4. Clause 5 presents the third, design phase using XML schemas (tML POTS schemas) for POTS Service Test functions and services. The design phase is a result of analyzing ATIS-0300262 data definitions shown in Informative Annex A. The general architecture for service level test is based on the “uncontrolled test” formalism in X.745. The test action performer is the agent in the managing system that performs the test based on the request from the managing system. Note that there is no coupling to a transport protocol. While mapping to a transport protocol is required for interoperability, it is outside the scope of this document, and should be addressed by each pair of bonded companies as a joint implementation issue.

Informative Annex A defines an information model using GDMO (ITU-T Recommendation X.722) and ASN.1 (ITU-T Recommendation X.680) for the TMN X-interface (ITU-T Recommendation M.3010) to support the service test function. These clauses are applicable when Common Management Information Service Element (CMISE) is used for conveying inquiry information across an OS-OS interactive interface.

Informative Annex B specifies a functionally equivalent interface using the Common Object Request Broker Architecture (CORBA)/Interface Definition Language (IDL), as defined in *The Common Object Request Broker Architecture and Specification, Revision 2.2, Object Management Group*.

1.3 Purpose

The purpose of this document is to provide an XML schema interface (tML ServiceTest interface) for uncontrolled POTS Service Test function specified in clause 3.

1.4 Abbreviations

For the purposes of this standard, the following abbreviations are used.

ANSI	American National Standards Institute
ASN.1	Abstract Syntax Notation One
ATIS	Alliance for Telecommunications Industry Solutions
BER	Basic Encoding Rules
CER	Canonical Encoding Rules
CLF	Common Language Facility
CMIP	Common Management Information Protocol
CMIS	Common Management Information Service
CMISE	Common Management Information Service Element
CNM	Customer Network Management
CORBA	Common Object Request Broker Architecture
DER	Distinguished Encoding Rules
GDMO	Guidelines for the Definition of Managed Objects
GNIM	General Network Information Model
GNM	General Network Model
HTTP (S)	Hyper Text Transport Protocol (Secure Sockets)
IDL	Interface Definition Language
MCS	Management Conformance Summary
MIDS	Management Information Definition Statement
MOCS	Managed Object Conformance Statement
MORT	Managed Object Referring to Test
MRCS	Management Relationship Conformance Statement
OAM&P	Operations, Administration, Maintenance & Provisioning
OBF	Ordering and Billing Forum
OMG	Object Management Group
OSI	Open Systems Interconnection
PDU	Protocol Data Unit
POTS	Plain Old Telephone Service
ROSE	Remote Operations Service Element
SOAP	Simple Object Access Model
STF	Service Test Function
SWC	Serving Wire Center
TMN	Telecommunications Management Network
tML	telecommunications Markup Language
W3C	World Wide Web Consortium
XML	Extensible Markup Language

1.5 Conventions Used in This Document

The reader should be aware of the convention used in this document that all XML Schema -.xsd- will be shown in `Courier New 9 point font`.

The XML Schema version used is < http://www.w3.org/2001/XMLSchema >, for the W3C's May 2, 2001, XML Schema Recommendation.

2 References

The following standards contain provisions which, through reference in this text, constitute provisions of this American National Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this American National Standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below.

2.1 Normative References

[1] *Extensible Markup Language (XML) 1.0, Fifth Edition*, Tim Bray, *et al*., eds,. W3C.[1]

[2] *Namespaces in XML 1.0, Third Edition*, Tim Bray, *et al*., eds,. W3C.[2]

[3] *XML Schema Part 0: Primer Second Edition*. David C. Fallside, *et al*., eds,. W3C Recommendation, latest version.[3]

[4] *XML Schema Part 1: Structures Second Edition*. Henry Thompson *et al*., eds,. W3C Recommendation, latest version.[4]

[5] *XML Schema Part 2: Datatypes Second Edition.* Paul Biron *et al*., eds,. W3C Recommendation, latest version.[5]

[6] ITU-T Recommendation M.3020, *Management interface specification methodology*.[6]

[7] ITU-T Recommendation M.3030, *Telecommunications Markup Language (tML) framework*.[6]

[8] ITU-T Recommendation M.3031, *Guidelines for Implementation Conformance Statement proformas for tML schemas*.[6]

[9] ITU-T Recommendation X.745 | ISO/IEC 10164-12 (E), *Data Network and Open System Communications – OSI Management – Systems Management: Test Management Function*.[6]

[10] ITU-T Recommendation X.680 | ISO/IEC 8824-1, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation (plus Amendments and Corrigendum)*.[6]

[11] ITU-T Recommendation M.3400, *Maintenance: Telecommunications Management Network, TMN Management Functions*.[6]

2.2 Informative References

2.2.1 American National Standards

ATIS-0300208, *Operations, Administration, Maintenance, and Provisioning (OAM&P) - Upper-layer Protocols for*

[1] See < http://www.w3.org/TR/REC-xml >.

[2] See < http://www.w3.org/TR/REC-xml-names/ >.

[3] See < http://www.w3.org/TR/xmlschema-0/ >.

[4] See < http://www.w3.org/TR/xmlschema-1/ >.

[5] See < http://www.w3.org/TR/xmlschema-2/ >.

[6] This document is available from the International Telecommunications Union. < http://www.itu.int/ITU-T/ >

Telecommunication Management Network (TMN), Interfaces Q3 and X Interfaces.[7]

ATIS-0300227, *Operations, Administrations, Maintenance and Provisioning (OAM&P) - Extension to Generic Network Information Model for Interfaces between Operations Systems across Jurisdictional Boundaries to Support Fault Management*.[8]

ATIS-0300227.a, *Supplement to Operations, Administrations, Maintenance and Provisioning (OAM&P) – Interfaces Between Operations Systems Across Jurisdictional Boundaries to Support Fault Management (Trouble Administration)*.[9]

ATIS-0300228, *OAM&P - Services for Interfaces between Operations Systems across Jurisdictional Boundaries to Support Fault Management (Trouble Administration)*.[10]

ATIS-0300240, *Operations, Administration, Maintenance, and Provisioning (OAM&P) - Generic Network Information Model for Interfaces between Operations Systems and Network Elements*.[11]

2.2.2 Other Standards

ITU-T Recommendation M.3010, *Principles for a Telecommunications Management Network*.[6]

ITU-T Recommendation M.3100, *Generic Network Information Model*.[6]

ITU-T Recommendation X.210, *Information Technology - Open Systems Interconnection - Basic Reference Model; Conventions for the Definition of OSI Services*.[6]

ITU-T Recommendation X.680 | ISO/IEC 8824-1, *Information technology - Abstract Syntax Notation One (ASN.1): Specification of Basic Notation*.[6]

ITU-T Recommendation X.690, *Information Technology - ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER), and Distinguished Encoding Rules (DER)*.[6]

ITU-T Recommendation X.701 | ISO/IEC 10040, *Information Technology - Open Systems Interconnection - System Management Overview*.[6]

ITU-T Recommendation X.721 | ISO/IEC 10165-2, *Information Technology - Open Systems Interconnection - Structure of Management Information: Definition of Management Information*.[6]

ITU-T Recommendation X.722 | ISO/IEC 10165-4, *Information technology - Open Systems Interconnection - Structure of management information: Guidelines for the definition of managed objects*.[6]

ITU-T Recommendation X.730 | ISO/IEC 10164-1, *Information technology - Open Systems Interconnection - Systems Management: Object management function*.[6]

---

[7] This document is available from the Alliance for Telecommunications Industry Solutions (ATIS) at < https://www.atis.org/docstore/product.aspx?id=24609 >.

[8] This document is available from the Alliance for Telecommunications Industry Solutions (ATIS) at < https://www.atis.org/docstore/product.aspx?id=25583 >.

[9] This document is available from the Alliance for Telecommunications Industry Solutions (ATIS) at < https://www.atis.org/docstore/product.aspx?id=26775 >.

[10] This document is available from the Alliance for Telecommunications Industry Solutions (ATIS) at < https://www.atis.org/docstore/product.aspx?id=25680 >.

[11] This document is available from the Alliance for Telecommunications Industry Solutions (ATIS) at < https://www.atis.org/docstore/product.aspx?id=24712 >.

3 Definitions

3.1 Agent: As defined in ISO/IEC 10040, the System Management Overview.

3.2 Customer: User, leaser, or purchaser of a service (e.g., subscriber, end user).

3.3 End User: End user is the user of the service.

3.4 Facility: A facility is a DS1/DS3 service type between a customer-designated premise and a SWC where multiplexing is provided.

3.5 Jurisdiction: Refers to the functional separation of telecommunications networks. A jurisdiction is one of the following four types:

1. Local Exchange Carrier Network;
2. Interexchange Carrier Network;
3. End User Network; or
4. Some combination of the above.

3.6 Manager: As defined in ISO/IEC 10040, the System Management Overview.

3.7 Service: Any connection provided between two designated customer locations. There are several types of services: Common Language Facility (CLF) to an end-user premise (CLS), and from a customer premise to a switch or a node (CLS).

3.8 Service Customer: The service customer is anyone who chooses to use the OS-to-OS interface to do network management across jurisdictions to achieve control of telecommunications services (or resources) provided by a service provider.

3.9 Service Provider: The service provider is anyone who offers a standard interface to allow network management across jurisdictions of telecommunications services (or resources) they provide. The service provider acts in the agent role.

4 Functional Requirements

4.1 Service Test Function for POTS Services

The service customer must be able to request that the service provider perform a test on a POTS service. The request can only be for a test on a single POTS service. The request from the service customer contains the following information:

1. *The type of test to be performed*, which can be one of the following:
    a) full
    b) quick
    c) central office
    d) loop
    e) startTone
    f) stopTone
2. *The POTS service to test*. The service is identified directly by sending the name of the POTS service (i.e., telephone number). Lineshare services may be identified by the Serial circuit ID rather than by the POTS telephone number.

Upon receipt of a test request, the service provider will perform the test on the POTS service. If the service is currently in use, the service provider may notify the service customer that the service is in use and the service provider will be attempting the test again. The number of retries is up to the implementation. Alternatively, the service provider may return an error stating the service is in use and discontinue processing the request. The service

provider may return an error stating that the service customer is not authorized to request a test for this service. If a trouble report exists for the service, the service provider may return an error stating that testing cannot be performed while trouble repair is in progress.

When the test is completed the results will be returned to the service customer. The results contain the following:

1. The test results, which include:
   a) *A summary description of the test results*. For a startTone, insure that the summary indicates that tone has been placed on the TN specified as the Service Id in the Service Test Request; for a stopTone, insure that the summary indicates that the tone has been removed.
   b) *For a quick test*, include:
      i) Optionally, DC Signature results:
         – Tip to ground resistance.
         – Tip to ground voltage.
         – Ring to ground resistance.
         – Ring to ground voltage.
         – Tip to ring resistance.
      ii) Optionally, AC Signature results:
         – Number of ringers.
         – Tip to ground resistance.
         – Ring to ground resistance.
         – Tip to ring resistance.
      iii) Optionally, balance results:
         – Capacitive (percent).
         – Longitudinal (decibels).
      iv) Optionally, loop results, one of the following:
         – Total loop.
         – Open distance from the central office.
   c) *For a central office test*:
      i) Optionally, DC Signature results:
         – Tip to ground resistance.
         – Tip to ground voltage.
         – Ring to ground resistance.
         – Ring to ground voltage.
         – Tip to ring resistance.
      ii) Optionally, AC Signature results:
         – Ringers.
         – Tip to ground resistance.
         – Ring to ground resistance.
         – Tip to ring resistance.
      iii) Optionally, Balance results:
         – Capacitive (percent).
         – Longitudinal (decibels).
      iv) Optionally, central office results:
         – Line circuit status (character string).

- Dial tone status (character string).

d) *For a loop test*:
   i) Optionally, DC Signature results:
      - Tip to ground resistance.
      - Tip to ground voltage.
      - Ring to ground resistance.
      - Ring to ground voltage.
      - Tip to ring resistance.
   ii) Optionally, AC Signature results:
      - Number of ringers.
      - Tip to ground resistance.
      - Ring to ground resistance.
      - Tip to ring resistance.
   iii) Optionally, Balance results:
      - Capacitive (percent).
      - Longitudinal (decibels).
   iv) Optionally, central office results:
      - Line circuit status (character string).
      - Dial tone status (character string).
   e) *For a full test*, optionally include quick, central office, and loop results.

2. Optionally, if the test fails, a description of possible corrective actions.

4.2 Security Requirements

Access control should prevent unauthorized entities from requesting tests on services. Authentication of the entity requesting the test is required to ensure that access control can be performed.

5 POTS Service Test: UML & XML Representations

In developing this standard, the UML representation of data type is provided to facilitate future migration to a protocol neutral model conforming to the methodology in M.3020. This clause contains two steps for the management information using UML for mapping the data types from ASN.1 in the first step and defining the XML schema in the second step. Because the management information required for the analysis has been included in Informative Annex A and X.745, the approach taken here is to develop the UML representation for the management information in ASN.1 notation and convert it to an application protocol specific design using XML. The XML message definition is exchanged on the interface between service customer and service provider using a request/response protocol data unit.

The explanation of the approach for developing the tML POTS Service Test interface is detailed in different clauses of this document. Clause 4.1 defines the mapping rules between ASN.1 and UML for representing the management information in a protocol independent approach. Clause 4.2 specifies the rules for generating XML schema for POTS Service Test functions and services. The detailed class diagram for trouble administration services including Request and Response, as well as attribute types, will be presented in clause 4.3.

The tML Service Test interface consists of two categories based on the message exchange between service customer and service provider: *Request* and *Response*. A *request* represents the message transmitted from service customer to service provider initiated by service customer. A *response* corresponds to the message conveyed from service provider to service customer resulting from a Request. Request and response in the tML ServiceTest

interface were created to represent data exchanged for tML operations. Users of this standard should consult clause 3 above (as well as Informative Annex A) for detailed explanation of the behaviors of attributes in operations. The detailed class diagram for attribute types used in uncontrolled POTS service test is presented in clause 4.3.

This standard uses upper camel case (i.e., UpperCamelCase) for XML tag definition throughout the implementation of tML Service Test schemas.

5.1 Attribute Type Mapping Rules

This clause provides explanations on mapping rules for a given ASN.1 definition to a corresponding XML UML notation and XML schema. The syntaxes defined are for the types in Informative Annex A. In cases where multiple type definitions exist for the same base ASN.1 type, only one XML mapping rule is provided. The data structures are applicable to POTS service test even though other applications may find them useful

Table 5.1 - INTEGER Type

<table>
<tr><th>ASN.1 Notation</th><th>UML Model</th><th>Schema</th></tr>
<tr><td>Percent::= INTEGER(0..100)

Excerpt from X.680

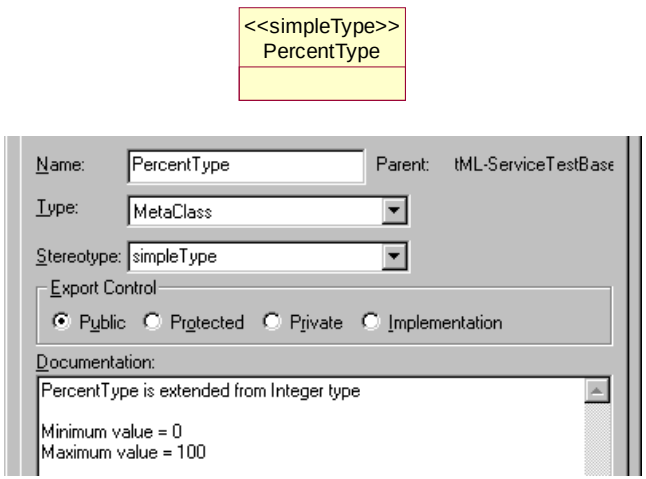
3.8.40 integer type: A simple type with distinguished values which are positive and negative whole numbers, including zero (as a single value).</td><td> 
</td><td><pre>
<simpleType name="PercentType">
    <restriction base="integer">
        <minInclusive value="0"/>
        <maxInclusive value="100"/>
    </restriction>
</simpleType>
</pre></td></tr>
</table>

Table 5.2 - INTEGER "Group"

ASN.1 Notation	UML Model	Schema

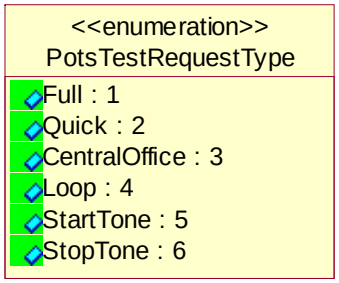
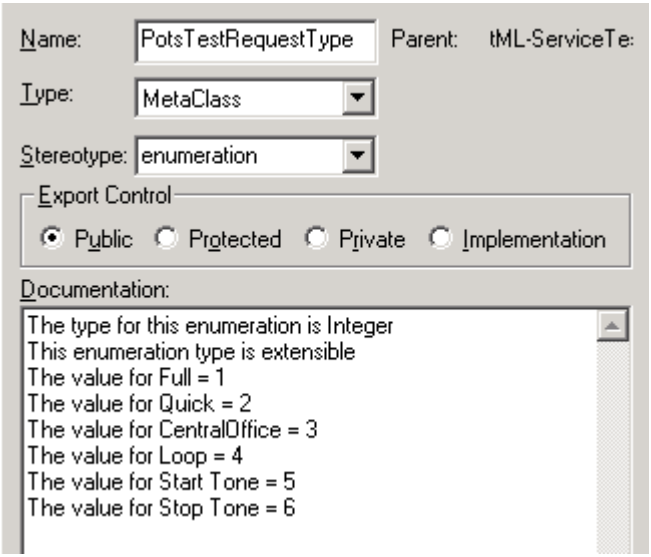
PotsTestRequest::= INTEGER {
full (1),
quick (2),
centraloffice (3),
loop (4),
startTone (5),
stopTone (6),
...
}
	
	
	 
	
		<simpleType name="PotsTestRequestType">
		<restriction base="integer">
		<annotation>
		<documentation>
		Full = 1
		Quick = 2
		Central Office = 3
		Loop = 4
		Start Tone = 5
		Stop Tone = 6
		</documentation>
		</annotation>
		<enumeration value="1"/>
		<enumeration value="2"/>
		<enumeration value="3"/>
		<enumeration value="4"/>
		<enumeration value="5"/>
		<enumeration value="6"/>
		</restriction>
		</simpleType>

Table 5.3 - SEQUENCE

ASN.1 Notation

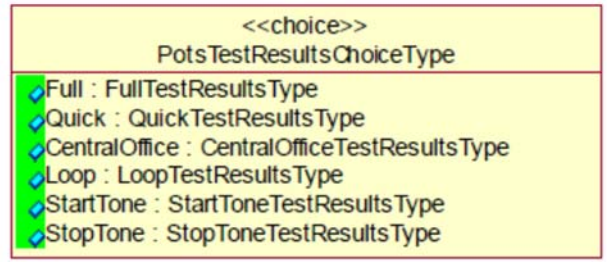
```
PotsTestResults::=SEQUENCE {
  summary      [0] GraphicString,
  results      [1] CHOICE {
    full              [0]          FullTestResults,
    quick             [1]          QuickTestResults,
    centralOffice     [2]          CentralOfficeTestResults,
    loop              [3]          LoopTestResults,
    startTone         [4]          StartToneTestResults,
    stopTone          [5]          StopToneTestResults,
    …
    } OPTIONAL
}
```

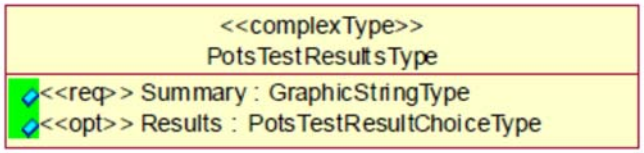
Schema

```
<complexType name="PotsTestResultsChoiceType">
    <choice>
        <element name="Full" type="tML-ServiceTestBase:FullTestResultsType"/>
        <element name="Quick" type="tML-ServiceTestBase:QuickTestResultsType"/>
        <element name="CentralOffice"
         type="tML-ServiceTestBase:CentralOfficeTestResultsType"/>
        <element name="Loop" type="tML-ServiceTestBase:LoopTestResultsType"/>
    <element name="StartTone" type="tML-ServiceTestBase:StartToneTestResultsType"/>
    <element name="StopTone" type="tML-ServiceTestBase:StopToneTestResultsType"/>
    </choice>
</complexType>

<complexType name="PotsTestResultsType" final="#all">
    <sequence>
        <element name="Summary" type="tML-TABase:GraphicStringType"/>
        <element name="Results" type="tML-ServiceTestBase:PotsTestResultsChoiceType"
         minOccurs="0"/>
    </sequence>
</complexType>
```

UML Model





Data Structure Diagram

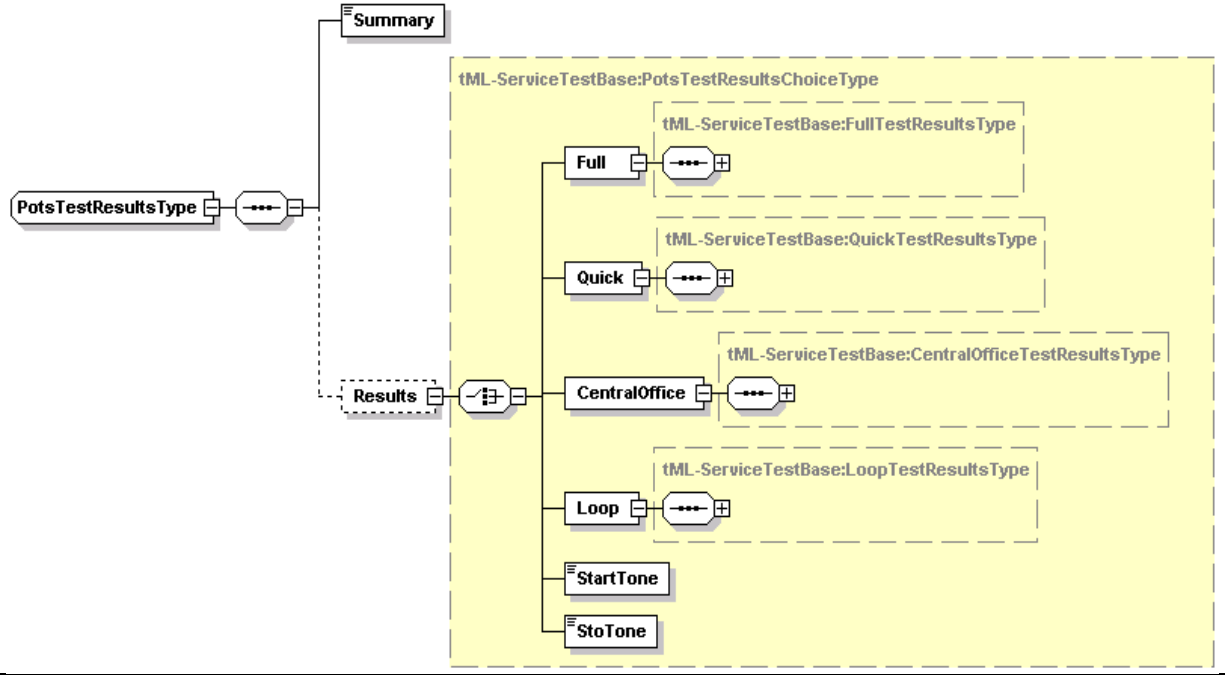
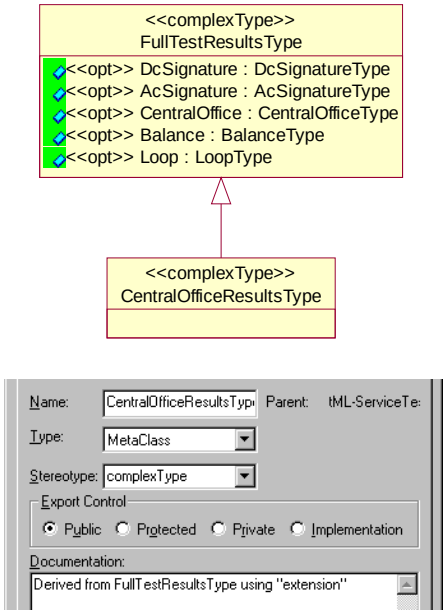


Table 5.4 - Inheritance by “extension”

ASN.1 Notation	UML Model	Schema

FullTestResults::=SEQUENCE {
dcSignature [1] DcSignature OPTIONAL,
acSignature [2] AcSignature OPTIONAL,
centralOffice [3] CentralOffice OPTIONAL,
balance [4] Balance OPTIONAL,
loop [5] Loop OPTIONAL,
…
}
 	
CentralOfficeTestResults::=SEQUENCE {
dcSignature [1] DcSignature OPTIONAL,
acSignature [2] AcSignature OPTIONAL,
centralOffice [3] CentralOffice OPTIONAL,
balance [4] Balance OPTIONAL,
loop [5] Loop OPTIONAL,
…
}
		<complexType name="FullTestResultType">
		<sequence>
		<element name="DcSignature"
		type="tML-ServiceTestBase:DcSignatureType" minOccurs="0"/>
		<element name="AcSignature"
		type="tML-ServiceTestBase:AcSignatureType" minOccurs="0"/>
		<element name="CentralOffice"
		type="tML-ServiceTestBase:CentralOfficeType" minOccurs="0"/>
		<element name="Balance"
		type="tML-ServiceTestBase:BalanceType" minOccurs="0"/>
		<element name="Loop"
		type="tML-ServiceTestBase:LoopType" minOccurs="0"/>
		</sequence>
		</complexType>
		
		<complexType name="CentralOfficeTestResultType">
		<complexContent>
		<extension
		base="tML-ServiceTestBase:FullTestResultType"/>
		</complexContent>
		</complexType>

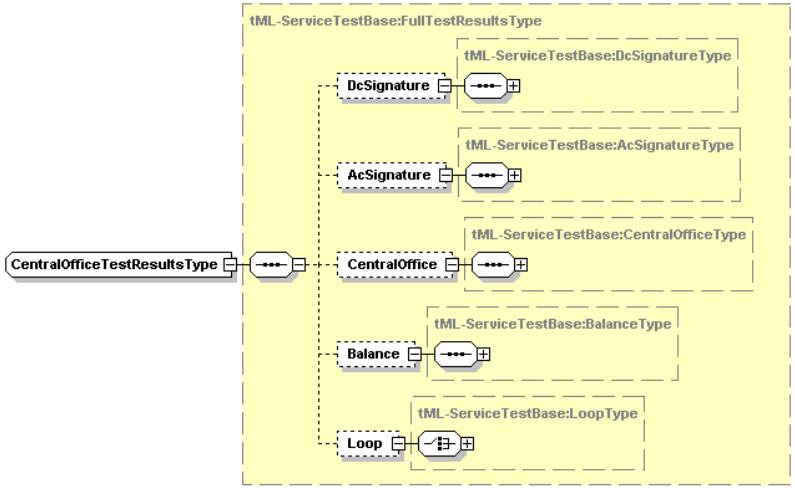
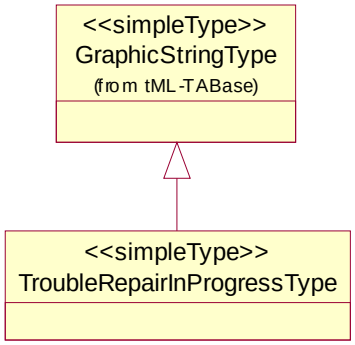
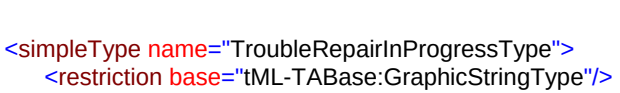
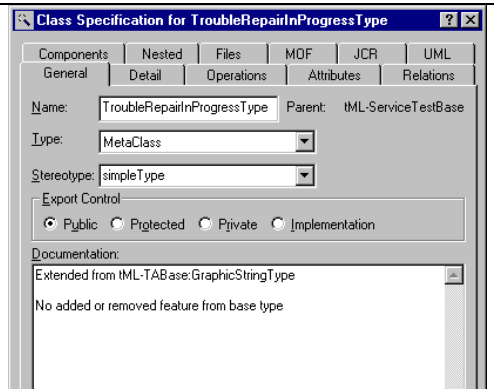
Data Structure Diagram


Table 5.5 - Parameter Mapping: troubleRepairInProgress

GDMO Definition	UML Model	Schema
6.8.3 troubleRepairInProgress **6.8.3.1 Description** The troubleRepairInProgress parameter is used to inform the test requester that the test request cannot be satisfied because the test performer does not allow test requests on service for which trouble has been reported. **6.8.3.2 Template** troubleRepairInProgress PARAMETER CONTEXT SPECIFIC-ERROR; WITH SYNTAX STF.TroubleRepairInProgress; BEHAVIOUR troubleRepairInProgressBehaviour BEHAVIOUR DEFINES AS “This parameter is used in the case where the test performer does not allow the test requester to request tests on services for which trouble has been reported.”;; REGISTERED AS {stfParameter 2};		


```
<simpleType name="TroubleRepairInProgressType">
    <restriction base="tML-TABase:GraphicStringType"/>
</simpleType>
```

ASN.1 Notation		
TroubleRepairInProgress::=GraphicString		

5.2 POTS Service Test Service Mapping Rules

The tML Service Test interface is developed based on the analysis on uncontrolled POTS Service Test service definitions in Informative Annex A.. The tML Service Test interface consists of two message categories: Request and Response.

The following table lists comparisons between the tML Service Test interface and POTS Service Test function and parameters specified in Informative Annex A.

Table 5.6 - Comparison between Service Test Function/Service and the tML Service Test Interface

POTS Test Functional Unit	Service Test Action and Parameters	XML Interface
Uncontrolled POTS Test	• TestRequestUncontrolled M_ACTION • potsUncontrolledTestRequest • potsUncontrolledTestResults	• POTSUncontrolledTestRequest/Response

Table 5.7 - Service Test Function for POTS Services Request/Response

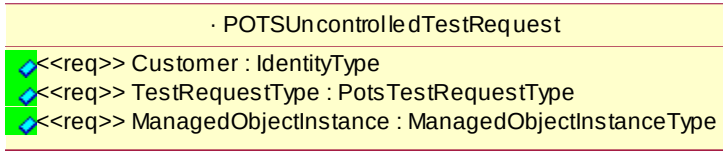
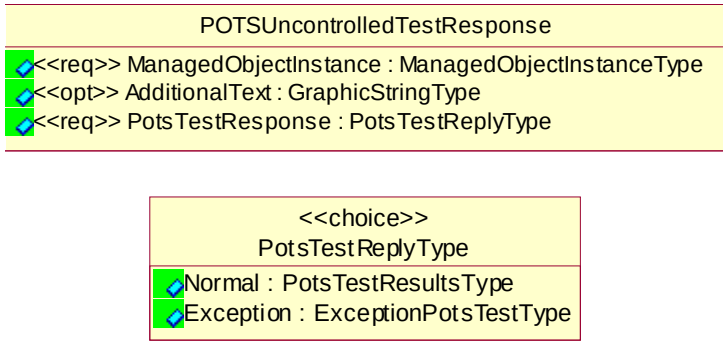
Excerpt from ATIS-0300262	UML Model (Request)	Schema (Request)
5.1.1 Service Test Function for POTS Services … The request can only be for a single POTS service. The request from the service customer contains the following information: 1. The type of test to be performed… 2. The POTS service to test… **6.1.1 Support of the Test Function for POTS Services** … The TestRequestUncontrolledInfo in the M-ACTION request will contain the following: 1) The testCategoryInformation… … **6.8.1 potsUncontrolledTestRequest** **6.8.1.1 Description** The potsUncontrolledTestRequest parameter is used to convey the type of test to performon a POTS service when using the UncontrolledTestRequest ACTION. **6.8.1.2 Template** potsUncontrolledTestRequest PARAMETER CONTEXT Test-ASN1Module. TestRequestUncontrolledInfo. testCategoryInformation; WITH SYNTAX STF.PotsTestRequest; BEHAVIOUR potsUncontrolledTestRequestBehaviour BEHAVIOUR DEFINED AS “This parameter is used to request the service test function to be performed on a POTS service”;; REGISTERED AS {stfParameter 1};		<element name="POTSUncontrolledTestRequest" type="tML-ServiceTest:POTSUncontrolledTestRequestType"/> <complexType name="POTSUncontrolledTestRequestType"> <sequence> <element name="Customer" type="tML-TABase:IdentityType"/> <element name="TestRequestType" type="tML-ServiceTestBase:PotsTestRequestType"/> <element name="ManagedObjectInstance" type="tML-TABase:ManagedObjectInstanceType"/> </sequence> </complexType>

Table 5.7 - Service Test Function for POTS Services Request/Response (con't)

Excerpt from ATIS-0300262	UML Model (Response)	Schema (Response)
5.1.1 Service Test Function for POTS Services … When the test is completed, the results will be returned to the service customer. The results contain the following: 1. The test results, which include:… 2. Optionally, if the test fails, a description of possible corrective actions. **6.1.1 Support of the Test Function for POTS Services** … The TestRequestUncontrolledInfo in the M-ACTION reply will contain the following: 1) testOutcome field will… … **6.8.2 potsUncontrolledTestResults** **6.8.2.1 Description** The potsUncontrolledTestResults parameter is used to convey the results of a test performed on a POTS service when using the UncontrolledTestRequest ACTION **6.8.2.2 Template** potsUncontrolledTestResults PARAMETER CONTEXT Test-ASN1Module. TestRequestUncontrolledResult. additionalInformation; WITH SYNTAX STF.PotsTestResults; BEHAVIOUR potsUncontrolledTestResultsBehaviour BEHAVIOUR DEFINED AS "This parameter contains the result of the service test function to be performed on a POTS service";; REGISTERED AS {stfParameter 1};		<element name="POTSUncontrolledTestResponse" type="tML-ServiceTest:POTSUncontrolledTestResponseType"/> <complexType name="POTSUncontrolledTestResponseType"> <sequence> <element name="ManagedObjectInstance" type="tML-TABase:ManagedObjectInstanceType"/> <element name="AdditionalText" type="tML-TABase:GraphicStringType" minOccurs="0"/> <element name="PotsTestResponse" type="tML-ServiceTestBase:PotsTestReplyType"/> </sequence> </complexType> <complexType name="PotsTestReplyType"> <choice> <element name="Normal" type="tML-ServiceTestBase:PotsTestResultsType"/> <element name="Exception" type="tML-ServiceTestBase:ExceptionPotsTestType"/> </choice> </complexType> <complexType name="ExceptionPotsTestType"> <complexContent> <extension base="tML-TABase:ExceptionGenericType"> <sequence> <element name="NotFound" type="tML-TABase:NotFoundType" minOccurs="0"/> <element name="MissingData" type="tML-TABase:MissingDataType" minOccurs="0"/> <element name="ServiceInUse" type="tML-ServiceTestBase:ServiceInUseType" minOccurs="0"/> <element name="TroubleRepairInProgress" type="tML-ServiceTestBase:TroubleRepairInProgressType" minOccurs="0"/> </sequence> </extension> </complexContent> </complexType>

5.3 UML Diagrams for tML Service Test Base

5.3.1 Attribute Types & Data Types Used in POTS Service Test

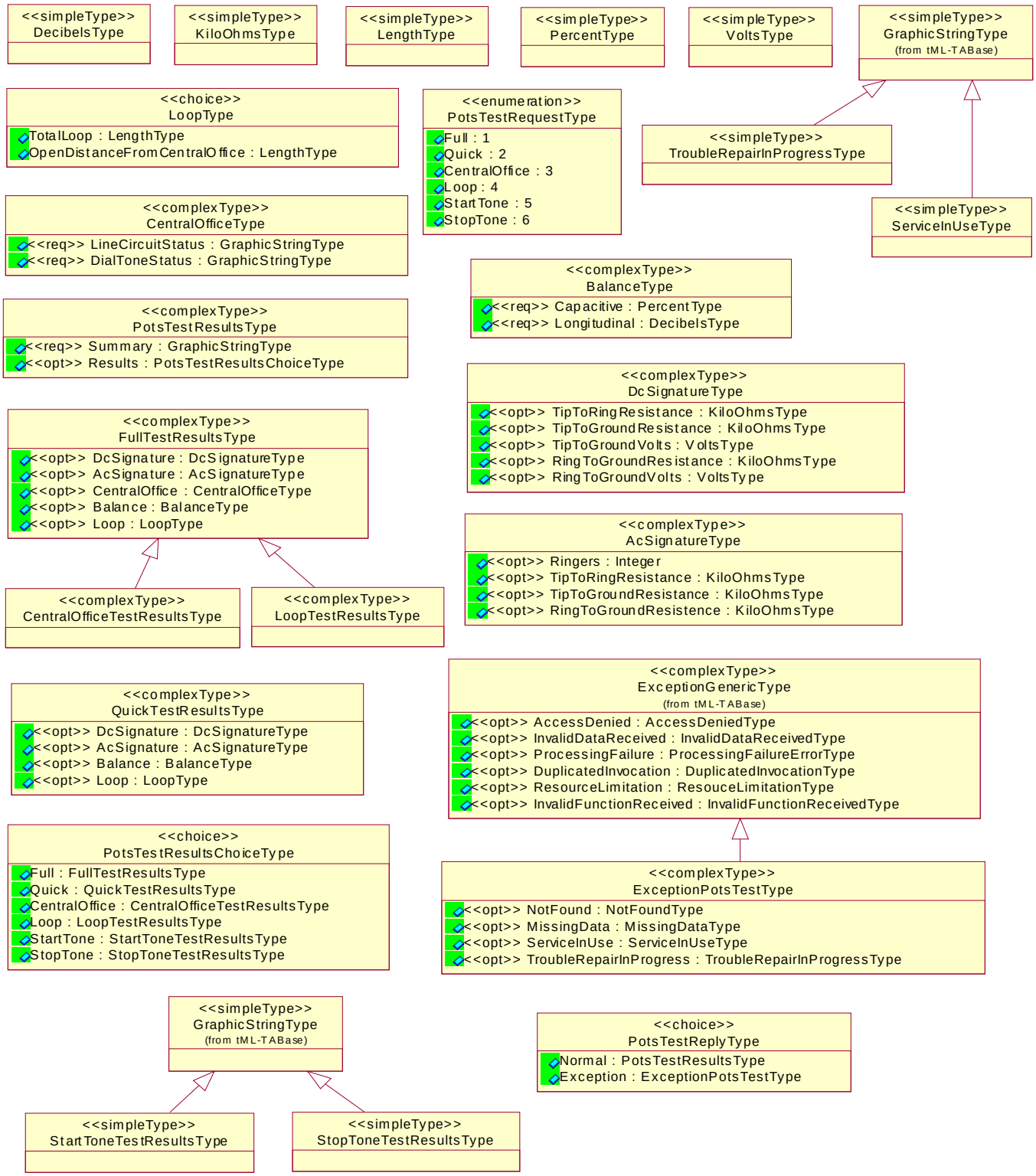


Figure 5.1 - Data Types Used in POTS Service Test

5.4 POTS Service Test XML Schemas

5.4.1 Overview of tML ServiceTest Schema Packages

5.4.1.1 tML ServiceTest Schema Packages

The suggested tML Service Test schema packages dependency is the following:

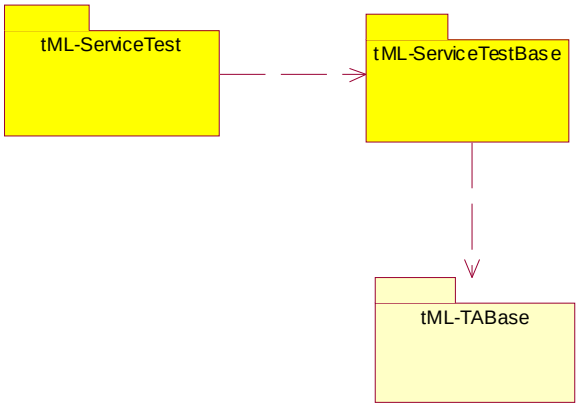


Figure 5.2 - Data Types Used in POTS Service Test

The impact of the dependencies in tML Service Test and tML TABase Schemas is shown in the following subclauses.

5.4.1.2 tML-ServiceTest Package

```
<?xml version="1.0" encoding="UTF-8"?>
<schema
    targetNamespace="http://www.ansi.org/tML/ServiceTest/POTS/tML-ServiceTest"
    xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:tML="urn:itu.int/tML/tMLSchemaMetadata"
    xmlns:tML-TABase="http://www.ansi.org/tML/TA/tML-TABase"
    xmlns:tML-ServiceTestBase="http://www.ansi.org/tML/ServiceTest/POTS/tML-ServiceTestBase" xmlns:tML-
    ServiceTest="http://www.ansi.org/tML/ServiceTest/POTS/tML-ServiceTest" elementFormDefault="qualified"
    attributeFormDefault="unqualified"
    version="1.0 2002/08/05 10:54:08"
    id="tML-ServiceTest.xsd"
    xml:lang="en"
>

    <annotation>
        <documentation>
            <tML:SchemaMetadata>
                <tML:OriginalAuthor>Wu Liu</tML:OriginalAuthor>
                <tML:CreationDate>08-05-2002</tML:CreationDate>
                <tML:Description>This schema contains uncontrolled POTS Service Test service
                </tML:Description>
                <tML:Source> ATIS-0300262.2007</tML:Source>
                <tML:SchemaHistory/>
            </tML:SchemaMetadata>
        </documentation>
```

```
    </annotation>

    <import namespace="http://www.ansi.org/tML/ServiceTest/POTS/tML-ServiceTestBase"
            schemaLocation="tML-ServiceTestBase.xsd"/>

    <!—detailed in clause 4.2 -->

</schema>
```

5.4.1.3 tML-ServiceTestBase Package

```
<?xml version="1.0" encoding="UTF-8"?>
<schema
    targetNamespace="http://www.ansi.org/tML/ServiceTest/POTS/tML-ServiceTestBase"
    xmlns:tML-ServiceTestBase="http://www.ansi.org/tML/ServiceTest/POTS/tML-ServiceTestBase"
    xmlns:tML-TABase="http://www.ansi.org/tML/TA/tML-TABase"
    xmlns:tML="urn:itu.int/tML/tMLSchemaMetadata"
    xmlns="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified"
    version="1.1 2004/04/14 14:07:18"
    id="tML-ServiceTestBase.xsd"
    xml:lang="en"
>
    <annotation>
        <documentation>
            <tML:SchemaMetadata>
                <tML:OriginalAuthor>Wu Liu</tML:OriginalAuthor>
                <tML:CreationDate>08-05-2002</tML:CreationDate>
                <tML:Description>This schema contains Attribute Types defined in ATIS- 0300262.2007 and related
                                 documents. It also provides data types used in POTS Service Test
                                 services (tML-ServiceTest schema)
                </tML:Description>
                <tML:Source> ATIS-0300262.2007</tML:Source>
                <tML:SchemaHistory/>
            </tML:SchemaMetadata>
        </documentation>
    </annotation>

    <import namespace="http://www.ansi.org/tML/TA/tML-TABase" schemaLocation="tML-TABase.xsd"/>

    <!—detailed in clause 4.3 -->

</schema>
```

5.5 Schema for tML-ServiceTest

5.5.1 Request Portion[12]

```
<element name="POTSUncontrolledTestRequest"
        type="tML-ServiceTest:POTSUncontrolledTestRequestType"/>

<complexType name="POTSUncontrolledTestRequestType">
   <sequence>
      <element name="Customer" type="tML-TABase:IdentityType"/>
      <element name="TestRequestType" type="tML-ServiceTestBase:PotsTestRequestType"/>
      <element name="ManagedObjectInstance" type="tML-TABase:ManagedObjectInstanceType"/>
   </sequence>
</complexType>
```

5.5.2 Response Portion

```
<element name="POTSUncontrolledTestResponse"
        type="tML-ServiceTest:POTSUncontrolledTestResponseType"/>

<complexType name="POTSUncontrolledTestResponseType">
   <sequence>
      <element name="ManagedObjectInstance" type="tML-TABase:ManagedObjectInstanceType"/>
      <element name="AdditionalText" type="tML-TABase:GraphicStringType" minOccurs="0"/>
      <element name="PotsTestResponse" type="tML-ServiceTestBase:PotsTestReplyType"/>
   </sequence>
</complexType>
```

5.6 Schema for tML – ServiceTestBase

5.6.1 Attribute Types

```
<simpleType name="DecibelsType">
   <restriction base="double"/>
</simpleType>
<simpleType name="KiloOhmsType">
   <restriction base="double"/>
</simpleType>
<simpleType name="LengthType">
   <restriction base="integer"/>
</simpleType>
<complexType name="LoopType">
   <choice>
      <element name="TotalLoop" type="tML-ServiceTestBase:LengthType"/>
      <element name="OpenDistanceFromCentralOffice" type="tML-ServiceTestBase:LengthType"/>
   </choice>
```

[12] *Message Identification* (commonly known as *Request Id*) is used to correlate a request and a corresponding response for a specific tML message exchanged between a Service Provider and a Service Customer. When this standard was first published as a trial use standard, transporting tML message under SOAP over HTTP(S) by certain Web Services imposed limitations on how the tML Header content could be constructed. Those limitations have led some Service Provider(s) and Service Customer(s) to move the Message Identification field from the tML Header into the tML message itself (as in tML Service Test request/response PDU where the identification is the first field in the message). Even with the Web Services software vendors progressing their products so that supporting a full tML Header in a SOAP message has become possible, those Service Provider(s) or Service Customer(s) who have already implemented tML Service Test interfaces using the Message Identification (Request Id) as the first field in tML Service Test request/response PDUs may continue do so and this practice would not be considered a violation of this standard.

```
</complexType>
<simpleType name="PercentType">
   <restriction base="integer">
      <minInclusive value="0"/>
      <maxInclusive value="100"/>
   </restriction>
</simpleType>
<simpleType name="VoltsType">
   <restriction base="double"/>
</simpleType>
<simpleType name="TroubleRepairInProgressType">
   <restriction base="tML-TABase:GraphicStringType"/>
</simpleType>
<simpleType name="ServiceInUseType">
   <restriction base="tML-TABase:GraphicStringType"/>
</simpleType>
<complexType name="AcSignatureType" final="#all">
   <sequence>
      <element name="Ringers" type="integer" minOccurs="0"/>
      <element name="TipToRingResistance" type="tML-ServiceTestBase:KiloOhmsType"
               minOccurs="0"/>
      <element name="TipToGroundResistance" type="tML-ServiceTestBase:KiloOhmsType"
               minOccurs="0"/>
      <element name="RingToGroundResistance" type="tML-ServiceTestBase:KiloOhmsType"
               minOccurs="0"/>
   </sequence>
</complexType>
<complexType name="DcSignatureType" final="#all">
   <sequence>
      <element name="TipToRingResistance" type="tML-ServiceTestBase:KiloOhmsType"
               minOccurs="0"/>
      <element name="TipToGroundResistance" type="tML-ServiceTestBase:KiloOhmsType"
               minOccurs="0"/>
      <element name="TipToGroundVolts" type="tML-ServiceTestBase:VoltsType" minOccurs="0"/>
      <element name="RingToGroundResistance" type="tML-ServiceTestBase:KiloOhmsType"
               minOccurs="0"/>
      <element name="RingToGroundVolts" type="tML-ServiceTestBase:VoltsType" minOccurs="0"/>
   </sequence>
</complexType>
<complexType name="BalanceType" final="#all">
   <sequence>
      <element name="Capacitive" type="tML-ServiceTestBase:PercentType"/>
      <element name="Longitudinal" type="tML-ServiceTestBase:DecibelsType"/>
   </sequence>
</complexType>
<complexType name="CentralOfficeType" final="#all">
   <sequence>
      <element name="LineCircuitStatus" type="tML-TABase:GraphicStringType"/>
      <element name="DialToneStatus" type="tML-TABase:GraphicStringType"/>
   </sequence>
</complexType>
<complexType name="FullTestResultsType">
   <sequence>
      <element name="DcSignature" type="tML-ServiceTestBase:DcSignatureType" minOccurs="0"/>
      <element name="AcSignature" type="tML-ServiceTestBase:AcSignatureType" minOccurs="0"/>
      <element name="CentralOffice" type="tML-ServiceTestBase:CentralOfficeType"
               minOccurs="0"/>
      <element name="Balance" type="tML-ServiceTestBase:BalanceType" minOccurs="0"/>
      <element name="Loop" type="tML-ServiceTestBase:LoopType" minOccurs="0"/>
   </sequence>
</complexType>
<complexType name="CentralOfficeTestResultsType">
   <complexContent>
      <extension base="tML-ServiceTestBase:FullTestResultsType"/>
   </complexContent>
```

```
</complexType>
<complexType name="LoopTestResultsType" final="extension">
   <complexContent>
      <extension base="tML-ServiceTestBase:FullTestResultsType"/>
   </complexContent>
</complexType>
<complexType name="QuickTestResultsType" final="extension">
   <sequence>
      <element name="DcSignature" type="tML-ServiceTestBase:DcSignatureType" minOccurs="0"/>
      <element name="AcSignature" type="tML-ServiceTestBase:AcSignatureType" minOccurs="0"/>
      <element name="Balance" type="tML-ServiceTestBase:BalanceType" minOccurs="0"/>
      <element name="Loop" type="tML-ServiceTestBase:LoopType" minOccurs="0"/>
   </sequence>
</complexType>
<simpleType name="StartToneTestResultsType">
   <restriction base="tML-TABase:GraphicStringType"/>
</simpleType>
<simpleType name="StopToneTestResultsType">
   <restriction base="tML-TABase:GraphicStringType"/>
</simpleType>
<simpleType name="PotsTestRequestType">
   <restriction base="integer">
      <enumeration value="1"/>
      <enumeration value="2"/>
      <enumeration value="3"/>
      <enumeration value="4"/>
      <enumeration value="5"/>
      <enumeration value="6"/>
   </restriction>
</simpleType>
<complexType name="PotsTestResultsChoiceType">
   <choice>
      <element name="Full" type="tML-ServiceTestBase:FullTestResultsType"/>
      <element name="Quick" type="tML-ServiceTestBase:QuickTestResultsType"/>
      <element name="CentralOffice" type="tML-ServiceTestBase:CentralOfficeTestResultsType"/>
      <element name="Loop" type="tML-ServiceTestBase:LoopTestResultsType"/>
      <element name="StartTone" type="tML-ServiceTestBase:StartToneTestResultsType"/>
      <element name="StopTone" type="tML-ServiceTestBase:StopToneTestResultsType"/>
   </choice>
</complexType>
<complexType name="PotsTestResultsType" final="#all">
   <sequence>
      <element name="Summary" type="tML-TABase:GraphicStringType"/>
      <element name="Results" type="tML-ServiceTestBase:PotsTestResultsChoiceType"
               minOccurs="0"/>
   </sequence>
</complexType>
<complexType name="PotsTestReplyType">
   <choice>
      <element name="Normal" type="tML-ServiceTestBase:PotsTestResultsType"/>
      <element name="Exception" type="tML-ServiceTestBase:ExceptionPotsTestType"/>
   </choice>
</complexType>
<complexType name="ExceptionPotsTestType">
   <complexContent>
      <extension base="tML-TABase:ExceptionGenericType">
         <sequence>
            <element name="NotFound" type="tML-TABase:NotFoundType" minOccurs="0"/>
            <element name="MissingData" type="tML-TABase:MissingDataType" minOccurs="0"/>
            <element name="ServiceInUse" type="tML-ServiceTestBase:ServiceInUseType"
                     minOccurs="0"/>
            <element name="TroubleRepairInProgress"
                     type="tML-ServiceTestBase:TroubleRepairInProgressType" minOccurs="0"/>
         </sequence>
      </extension>
```

```
   </complexContent>
</complexType>
```

Annex A
(informative)

A CMIP (ASN.1) Model & Accompanying Documentation from ATIS-0300262-2007

This informative annex contains the most recent CMIP version of the POTS Service Test Function. It is the CMIP equivalent of the modeling shown in the body of this document.

A.1 Object Model

The object model in this standard supports the Test Function for services, which is the interactive data transfer of service test information between service providers and service customers.

This object model pertains to the TMN X-interface supporting the management at service level (defined in ITU-T Recommendation M.3010). The model uses and is consistent with existing OSI system management recommendations (ITU-T Recommendation X.721, X.722, etc.), generic network information model recommendations/standards (ITU-T Recommendation M.3100, ATIS-0300240) and OAM&P standards for electronic bonding applications (e.g., ATIS-0300227). Some of the managed object classes defined here are derived from the object classes in the above standards and recommendations.

A.1.1 Object Model Overview

The object model described in this standard is a specialization of the object model defined in ITU-T Recommendation X.745 for the testing of specific services across the X interface. The testActionPerformer managed object class in ITU-T Recommendation X.745 is extended to specify specific tests for particular service types. The serviceTestActionPerformer managed object class defined in this standard provides a consistent name binding for all subclasses. To define parameters specific for performing tests on a service type, a new managed object is derived from the serviceTestActionPerformer. The parameters consistent with the object model in ITU-T Recommendation X.745 are added to the derived managed object class.

The policy of creating an instance of a derived class of the serviceTestActionPerformer is left to the service provider implementation. These managed objects are created inherently by the agent and the number of these objects and their selection of name bindings are outside the scope of this standard.

A.1.1.1 Support of the Test Function for POTS Services

In order to perform the service test function for a POTS service, the test request uncontrolled service defined in ITU-T Recommendation X.745 is used. A testRequestUncontrolled M_ACTION is invoked on a potsTestActionPerformer managed object. This managed object can be created for a specific test or be reused for multiple tests allowing different implementation strategies.

The TestRequestUncontrolledInfo in the M-ACTION request will contain the following:

1. The testCategoryInformation field will contain the information defined in the potsUncontrolledTestRequest parameter. The potsUncontrolledTestRequest parameter identifies the test to be performed.
2. The testSessionId, timeoutPeriod, and associatedObjects fields are not used.
3. If the POTS service to be tested is identified by the service name, the service name will be in the toBeTestedMORTs field.

The TestRequestUncontrolledResult contained in the M-ACTION reply will contain the following:

1. testOutcome field will optionally contain the test outcome as defined in ITU-T Recommendation X.745. This information may be redundant with the specific results specified in the additional Information field.
2. The service that was tested will be identified in the Managed Object Referring to Test (MORT) field as defined in ITU-T Recommendation X.745 | ISO/IEC 10164-12.
3. The proposedRepairActions field will not be used (see item 4).
4. If the test identified a problem and the service provider wants to notify the service customer of the proposed repair actions, the additionalText field will contain a description of the proposed repair actions.
5. The additionalInformation field will contain the information defined in the potsUncontrolledTestResult parameter. The potsUncontrolledTestResult parameter specifies the PotsTestResult syntax which contains the result information for the test selected.

If the service is not available for testing, the mORTNotAvailable parameter (defined in ITU-T Recommendation X.745) will be returned. If the test requester does not have the authority to test the service, the CMISE accessDenied error will be returned. If the test performer does not allow test requests on services for which trouble has been reported, the TroubleRepairInProgress parameter will be returned.

A.1.1.2 Support of Security Requirements

This standard assumes authentication and access control to be performed as defined in clause 9.

A.1.2 Inheritance Hierarchy

Some object classes are defined as "subclasses of another object". A subclass inherits the properties (attributes, events, actions, and behaviors) of the superclass from which it is derived. An inheritance hierarchy represents the succession of inheritance relationships. The inheritance hierarchy of this document is illustrated in Figure A.1. The boxes represent managed object classes, and the connecting lines represent the subclassing relationships. The object class "top" is at the apex of the classification hierarchy.

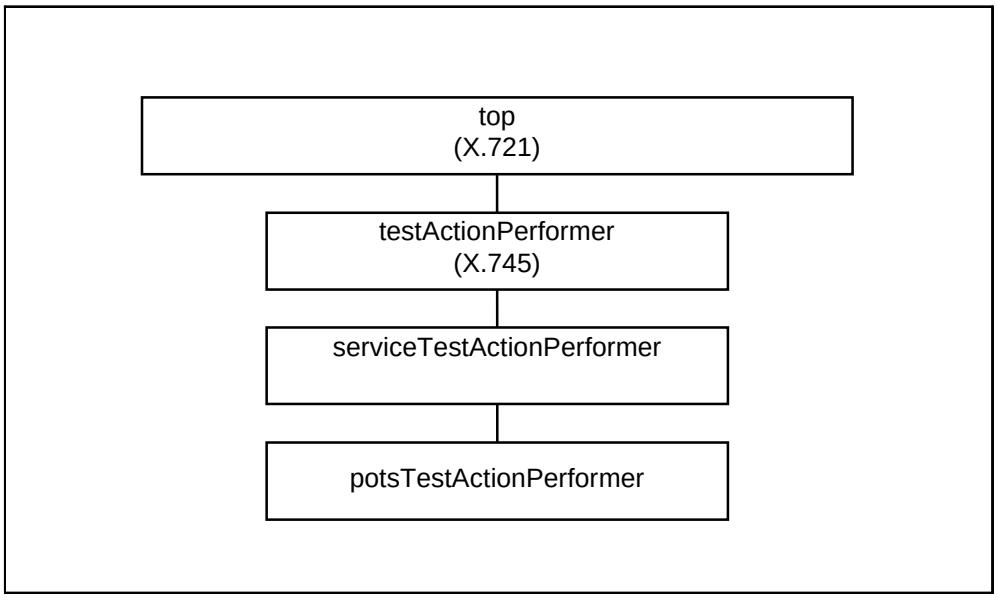


Figure A.1 - Inheritance Hierarchy

A.1.3 Naming Tree (Containment) & Pointer Relationships

Figure A.2 illustrates the naming and pointer relationships.

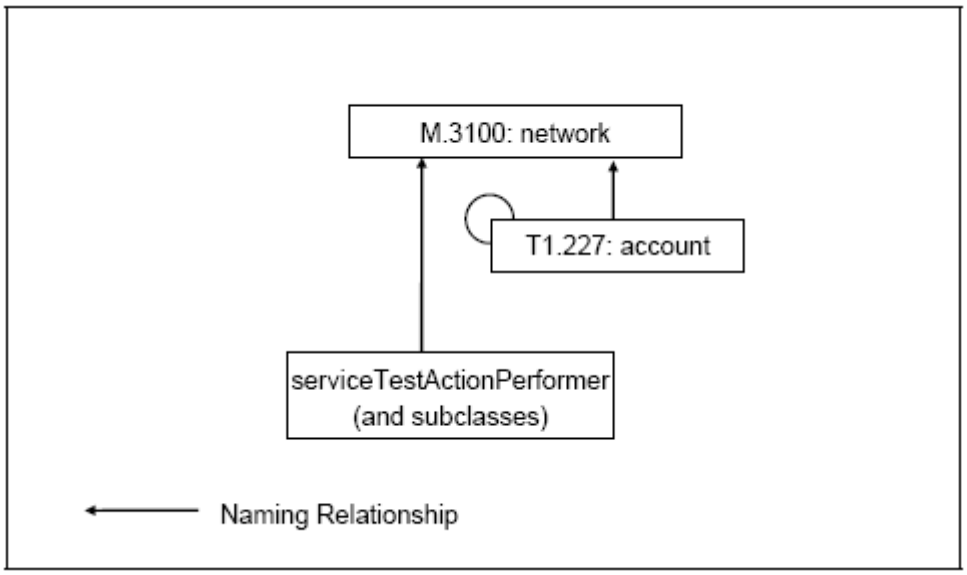


Figure A.2 - Naming Relationships

A.1.4 Managed Object Classes

A.1.4.1 serviceTestActionPerformer

A.1.4.1.1 Description

The serviceTestActionPerformer is an abstract base class derived from the TestActionPerformer managed object class defined in ITU-T Recommendation X.745. The main purpose of this class is to allow all classes derived from it to have a common name binding and allow the use of the createDeleteNotification package.

A.1.4.1.2 Template

```
serviceTestActionPerformer  MANAGED OBJECT CLASS
DERIVED FROM "Rec. X.745|ISO/IEC 10164-12": testActionPerformer;
CHARACTERIZED BY
serviceTestActionPerformerPkg  PACKAGE
       BEHAVIOUR
              serviceTestActionPerformerBehaviour BEHAVIOUR
              DEFINED AS
              "The serviceTestActionPerformer is an abstract base class designed to allow all
              service related testActionPerformers derived from it to have a common set of name
              bindings and to use the createDeleteNotificationPackage.”;;
       ;;
CONDITIONAL PACKAGES
       “Rec.M.3100”:createDeleteNotificationsPackage
              PRESENT IF “an instance supports it”;
REGISTERED AS {stfObjectClass 1};
```

A.1.4.2 potsTestActionPerformer

A.1.4.2.1 Description

The potsTestActionPerformer is a subclass of the serviceTestActionPerformer managed object class defined in the previous subclause of this standard. The potsTestActionPerformer adds the parameter to perform tests on POTS services.

A.1.4.2.2 Template

```
potsTestActionPerformer  MANAGED OBJECT CLASS
DERIVED FROM serviceTestActionPerformer;
CHARACTERIZED BY
potsTestActionPerformerPkg  PACKAGE
       BEHAVIOUR
              potsTestActionPerformerBehaviour BEHAVIOUR
              DEFINED AS
              "The potsTestActionPerformer is an object class designed to facilitate the testing
              of POTS services. For additional behaviour please refer to section 6.1.1”;;
       ACTIONS
              testRequestUncontrolledAction    potsUncontrolledTestRequest
              potsUncontrolledTestResults troubleRepairInProgress;
       ;;
REGISTERED AS {stfObjectClass 2};
```

A.1.5 Packages

Nothing defined.

A.1.6 Actions

Nothing defined.

A.1.7 Notifications

Nothing defined.

A.1.8 Parameters

A.1.8.1 potsUncontrolledTestRequest

A.1.8.1.1 Description

The potsUncontrolledTestRequest parameter is used to convey the type of test to perform on a POTS service when using the UncontrolledTestRequest ACTION.

A.1.8.1.2 Template

```
potsUncontrolledTestRequest PARAMETER
CONTEXT Test-ASN1Module.TestRequestUncontrolledInfo.testCategoryInformation;
```

```
       WITH SYNTAX STF.PotsTestRequest;
       BEHAVIOUR
potsUncontrolledTestRequestBehaviour BEHAVIOUR
DEFINED AS
"This parameter is used to request the service test function to be performed on a POTS
service";;
REGISTERED AS {stfParameter    1};
```

A.1.8.2 potsUncontrolledTestResults

A.1.8.2.1 Description

The potsUncontrolledTestResults parameter is used to convey the results of a test performed on a POTS service when using the UncontrolledTestRequest ACTION.

A.1.8.2.2 Template

```
potsUncontrolledTestResults PARAMETER
CONTEXT
Test-ASN1Module.TestRequestUncontrolledResult.additionalInformation;
       WITH SYNTAX STF.PotsTestResults;
       BEHAVIOUR
       potsUncontrolledTestResultsBehaviour BEHAVIOUR
       DEFINED AS
       "This parameter contains the results of the service test function for a POTS service.";;
REGISTERED AS {stfParameter    2};
```

A.1.8.3 troubleRepairInProgress

A.1.8.3.1 Description

The troubleRepairInProgress parameter is used to inform the test requester that the test request cannot be satisfied because the test performer does not allow test requests on service for which trouble has been reported.

A.1.8.3.2 Template

```
troubleRepairInProgress PARAMETER
CONTEXT SPECIFIC-ERROR;
WITH SYNTAX STF.TroubleRepairInProgress;
BEHAVIOUR
       troubleRepairInProgressBehaviour BEHAVIOUR
       DEFINED AS
              "This parameter is used in the case where the test performer does not allow the
              test requester to request tests on services for which trouble has been
              reported.";;
REGISTERED AS {stfParameter    2};
```

A.1.9 Attributes

Nothing defined.

A.1.10 Name Bindings

A.1.10.1 serviceTestActionPerformer-account

```
serviceTestActionPerformer-account NAME BINDING
       SUBORDINATE OBJECT CLASS serviceTestActionPerformer AND SUBCLASSES;
       NAMED BY
       SUPERIOR OBJECT CLASS "T1.227":account;
       WITH ATTRIBUTE testActionPerformerId;
REGISTERED AS {stfNameBinding 1};
```

A.1.10.2 serviceTestActionPerformer-network

```
serviceTestActionPerformer-network NAME BINDING
       SUBORDINATE OBJECT CLASS serviceTestActionPerformer AND SUBCLASSES;
       NAMED BY
       SUPERIOR OBJECT CLASS “Rec. M.3100”:network;
       WITH ATTRIBUTE testActionPerformerId;
REGISTERED AS {stfNameBinding 2};
```

A.1.11 Extensibility Rules

In accordance with ISO 8824-1, the productions that are of extensible types are to be indicated by including ellipsis (...) following in their type descriptions.

The following types will be indicated as being extensible:

- ENUMERATED
- named INTEGER
- named BIT STRING
- SET
- SEQUENCE
- CHOICE

Under the rules of extensibility, new enumerations (for ENUMERATED types), new bit name assignments (for named BIT STRING types), new named numbers (for named INTEGER types), and new elements (for SET, SEQUENCE, and CHOICE types) may be added in future versions of this standard.

When processing information in a System Management Application Protocol (SMAP) PDU, the accepting SMAP-machine shall issue RORJapdu (corresponding to the service RO-REJECT-U) with "mistypedResult" parameter for the following conditions:

- Enumerations not recognized.
- Unrecognized named numbers.
- Unrecognized named bits.
- Unrecognized tagged elements of sets, sequences, and choices.

A.1.12 Supporting Productions

```
STF {1.3.6.1.4.30613.2.1 stf(0)}
DEFINITIONS IMPLICIT TAGS ::=  BEGIN

-- EXPORTS Everything
```

```
IMPORTS

ObjectInstance
FROM CMIP-1 {joint-iso-ccitt(2) ms(9) cmip(1) modules(0) protocol(3)}

NameType
FROM ASN1DefinedTypesModule  { ccitt recommendation m(13) gnm(3100) informationModel(0)
asn1Modules(2)  asn1DefinedTypesModule(0) };

stfObjectClass OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) objectClass(3)}
stfPackage OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840 10055)
stf(0) package(4)}
stfParameter OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) parameter(5)}
stfNameBinding OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) nameBinding(6)}
stfAttribute OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) attribute(7)}
stfBehaviour OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) behaviour(8)}
stfAction OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840 10055)
stf(0) action(9)}
stfNotification OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) notification(10)}

-- The default value for this syntax is full.
PotsTestRequest ::= INTEGER {
       full               (1),
       quick              (2),
       centralOffice      (3),
       loop               (4),
       startTone          (5),
       stopTone           (6),
       ...
}
PotsTestResults ::= SEQUENCE {
       summary            [0] GraphicString,
       results            [1] CHOICE {
              full                [0]   FullTestResults,
              quick               [1]   QuickTestResults,
              centralOffice [2]   CentralOfficeTestResults,
              loop                [3]   LoopTestResults,
              startTone           [4]   StartToneTestResults,
              stopTone            [5]   StopToneTestResults,
              ...
       } OPTIONAL
}

FullTestResults ::= SEQUENCE {
dcSignature         [1] DcSignature           OPTIONAL,
acSignature         [2] AcSignature           OPTIONAL,
centralOffice       [3] CentralOffice         OPTIONAL,
balance             [4] Balance               OPTIONAL,
loop                [5] Loop                  OPTIONAL,
...
}
QuickTestResults ::= SEQUENCE {
       dcSignature         [1] DcSignature     OPTIONAL,
       acSignature         [2] AcSignature     OPTIONAL,
       balance             [3] Balance         OPTIONAL,
       loop                [4] Loop            OPTIONAL
}

CentralOfficeTestResults ::= SEQUENCE {
```

```
       dcSignature         [1] DcSignature           OPTIONAL,
       acSignature         [2] AcSignature           OPTIONAL,
       centralOffice [3] CentralOffice         OPTIONAL,
       balance             [4] Balance               OPTIONAL,
       loop                [5] Loop                  OPTIONAL,
...
}

LoopTestResults ::= SEQUENCE {
       dcSignature         [1] DcSignature           OPTIONAL,
       acSignature         [2] AcSignature           OPTIONAL,
       centralOffice [3] CentralOffice         OPTIONAL,
       balance             [4] Balance               OPTIONAL,
       loop                [5] Loop                  OPTIONAL
}

StartToneTestResults ::= GraphicString
StopToneTestResults ::= GraphicString

AcSignature ::= SEQUENCE {
       ringers                    [0] INTEGER         OPTIONAL,
       tipToRingResistance [1] KiloOhmns OPTIONAL,
       tipToGroundResistance      [2] KiloOhmns OPTIONAL,
       ringToGroundResistance     [4] KiloOhmns OPTIONAL
}

Balance ::= SEQUENCE {
       capacitive          [0] Percent,
       longitudinal [1] Decibels
}
CentralOffice ::= SEQUENCE {
       lineCircuitStatus   [0] GraphicString,
       dialToneStatus      [1] GraphicString
}
DcSignature ::= SEQUENCE {
       tipToRingResistance [0] KiloOhmns OPTIONAL,
       tipToGroundResistance      [1] KiloOhmns OPTIONAL,
       tipToGroundVolts           [2] Volts           OPTIONAL,
       ringToGroundResistance     [3] KiloOhmns OPTIONAL,
       ringToGroundVolts          [4] Volts           OPTIONAL
}
Decibels ::= INTEGER
KiloOhmns ::= INTEGER

Length ::= INTEGER

Loop ::= CHOICE {
       totalLoop                        [0] Length,
       openDistanceFromCentralOffice    [1] Length
}

Percent ::= INTEGER(0..100)

Volts ::= INTEGER

TroubleRepairInProgress ::= GraphicString

END  -- STF
```

A.2 Functional Units & Services

A.2.1 Table of Functional Units, Services, & Objects

Functional units allow negotiation for use of various services on an association (during association establishment). Table A.1 lists the STF functional units and corresponding services and object classes.

Table A.1 - Service Test Functional Units and Corresponding Services

Functional Unit	Services	Object Classes
POTS Test	UncontrolledTestRequest action	PotsTestActionPerformer

A.2.2 Service Definitions

The following services are defined in ITU-T Recommendation X.730:

- PT-GET

This subclause provides definitions for the following services:

- UncontrolledTestRequest action

The definition of each STF service in this standard includes a table which lists the parameters of its primitives. For a given primitive, the presence of each parameter is described by one of the following values:

M The parameter is mandatory.

(=) The value of the parameter is equal to the value of the parameter in the column to the left.

U Use of the parameter is a service-user option.

– The parameter is not present in the interaction.

C The parameter is conditionally present (the conditions are described by the text that describes the parameter).

The service definitions in 7.2 are described using the service definition conventions specified in ITU-T Recommendation X.210.

A.2.2.1 UncontrolledTestRequest Action Service to Support POTS Service Testing

This standard extends the UncontrolledTestRequest Action Service to support POTS service testing. The test request uncontrolled info and the test request uncontrolled response are extended in accordance with the following table.

Table A.2 - Test Request Uncontrolled Parameters for POTS Service Testing

Parameter Name	Req/Ind	Rsp/Conf
Test request uncontrolled info.	M	–
Test category information	M	–
Pots test request	M	–
Test session identifier	–	–

Parameter Name	Req/Ind	Rsp/Conf
Timeout period	–	–
Associated objects	C	–
To be tested MORT(s)	C	–
Test request uncontrolled response	–	C
Test outcome	–	C
MORT(s)	–	M
Proposed repair actions	–	–
Additional text	–	U
Additional information	–	M
Pots test results	–	M

A.2.3 Negotiation of Functional Units

This specification assigns the following object identifier value

```
{1.3.6.1.4.30613.2.1 functionalUnitPackage(1)}
```

as a value of the ASN.1 type FunctionalUnitPackagedId defined in ITU-T Recommendation X.701 | ISO/IEC 10040 to use for negotiating the following functional units:

1 POTS Test

where the number identifies the bit position assigned to the functional unit, and the name refers to the functional unit as defined in clause 7 of this model.

A.3 Application Service Elements & Application Context

The System Management application context defined in ITU-T Recommendation X.701 |I SO/IEC 10040 may be used.

A.4 Security

The actual security mechanisms for authentication and access control are outside the scope of this standard. However, the second Access Control syntax definition and corresponding registered abstract syntax ({iso(1) member-body(2) usa(840) ansi t1-228-1996(10016) specificAccessControlAbstractSyntax(4) encryptionMethod(1)}) defined in ATIS-0300228 may be used for the access control parameter in CMIPUserInfo of A-ASSOCIATE requests and responses and CMIP operations request PDUs. Other approaches to security are not precluded.

Annex B
(informative)

B CORBA/IDL Interface Design

B.1 Interface Definitions

B.1.1 PotsTestActionPerformer

PotsTestActionPerformer provides an interface for testing Plain Old Telephone Service (POTS). It is intended that the implementation of this interface be a service object. That is, only one instance provides the service for a particular implementation.

B.1.1.1 testRequestUncontrolled

Used to invoke a test on the specified POTS circuit.

```
void testRequestUncontrolled(
            in  RequestIdType               requestId,
            in  IdentityType                customer,
            in  PotsTestRequestType         request,
            out PotsTestResponseType        response,
                        inout TagValueListTypeOpt      appSpecificData
        )
        raises(
            FallBackReportingError,
            TroubleRepairInProgress,
            ServiceInUse,
            NotFound,
            AccessDenied,
            MissingData,
            InvalidDataReceived,
            ProcessingFailureError
        );
```

Parameters:

requestId: Unique ID identifying the client's request. Intended for tracking across the X interface.

Customer: Identifies the end user at the company, account, or user level.

Request: Holds the request information.

Reponse: Holds the test results.

appSpecificData: List of tags and associated values to be defined and employed on a per-implementation basis.

ReturnValues:

NONE.

Exceptions:

FallBackReportingError: Provider will not allow this request for reason indicated.

TroubleRepairInProgress: An open trouble report exists on the service.

ServiceInUse: Test failed because the service is currently in use.

NotFound: The specified POTS line wasn't found in the provider's network.

AccessDenied: The customer is not authorized to perform the requested transaction (e.g., the service on which the test was requested does not belong to the customer).

MissingData: An expected value was not provided.

InvalidDataReceived: The input parameters contain invalid data.

ProcessingFailureError: A system-related error (e.g., system timed out, resources exceeded occurred during processing).

B.2 IDL Modules

The following IDL are new to this document:

- ANSI T1 262 IDL Module

The following IDL modules are reused as defined in T1.227a-2001 and not included here:

- Base Module
- Customer Types Module:
- TimeTypes Module
- Location Types Module
- ANSI T1 227 IDL Module

```
ANSI T1 262 Module

/* This IDL code is meant to be stored in a file named "ansi_t1_262.idl"
 * located in the search path used by IDL compilers on your system.
 */

#ifndef ansi_t1_262_IDL
#define ansi_t1_262_IDL

#include <Base.idl>
#include <TimeTypes.idl>
#include <CustomerTypes.idl>
#include <ansi_t1_227.idl>

#pragma prefix "t1.org"

#define AccessDenied            Base::AccessDenied
#define FallBackReportingError  ansi_t1_227::FallBackReportingError
#define InvalidDataReceived     Base::InvalidData
#define MissingData             Base::MissingData
#define NotFound                Base::NotFound
#define ProcessingFailureError  Base::ProcessingFailureError

//
//
//  ansi_t1_262: Provides an interface for testing of Plain Old Telephone
//               Service (POTS). This interface is based on ANSI T1.262
//               (POTS Service Test Function).
```

```
//
//
module ansi_t1_262
{
    //-------------------------------------------------------
    // TYPES
    //-------------------------------------------------------

    typedef CustomerTypes::IdentityType   IdentityType;
    typedef Base::Null_t                  NullType;
    typedef Base::RequestId_t             RequestIdType;
    typedef Base::StringOpt_t             StringTypeOpt;
    typedef ansi_t1_227::ServiceIdType    ServiceIdType;
    typedef Base::TagValueListOpt_t       TagValueListTypeOpt;

    exception TroubleRepairInProgress {
        StringTypeOpt   additionalInformation;
    };

    exception ServiceInUse {
        StringTypeOpt   additionalInformation;
    };

    enum PotsTestTypeType {
        UnknownPotsTestType,
        FullTest,
        QuickTest,
        CentralOfficeTest,
        LoopTest,
         StartToneTest,
         StopToneTest
    };

    struct PotsTestRequestType {
        ServiceIdType                 serviceId;
        PotsTestTypeType              type;
    };

    typedef boolean RingersType;

    union RingersTypeOpt switch(boolean) {
      case TRUE: RingersType theValue;
    };

    typedef short KiloOhmsType;

    union KiloOhmsTypeOpt switch(boolean) {
      case TRUE: KiloOhmsType theValue;
    };

    typedef short VoltsType;

    union VoltsTypeOpt switch(boolean) {
      case TRUE: VoltsType theValue;
    };

    typedef short DecibelsType;

    union DecibelsTypeOpt switch(boolean) {
      case TRUE: DecibelsType theValue;
    };

    struct AcSignatureType {
        RingersTypeOpt                ringers;
        KiloOhmsTypeOpt               tipToRingResistance;
```

```
    KiloOhmsTypeOpt              tipToGroundResistance;
    KiloOhmsTypeOpt              ringToGroundResistance;
};

union AcSignatureTypeOpt switch(boolean) {
  case TRUE: AcSignatureType theValue;
};

struct BalanceType {
    long                    capacitive;
    DecibelsType              longitudinal;
};

union BalanceTypeOpt switch(boolean) {
  case TRUE: BalanceType theValue;
};

struct CentralOfficeType {
    string                lineCircuitStatus;
    string                dialToneStatus;
};

union CentralOfficeTypeOpt switch(boolean) {
  case TRUE: CentralOfficeType theValue;
};

enum LoopChoiceType {
    TotalLoopChoice,
    OpenDistanceFromCentralOfficeChoice
};

union LoopType switch(LoopChoiceType) {
  case TotalLoopChoice:
    long totalLoop;
  case OpenDistanceFromCentralOfficeChoice:
    long openDistanceFromCentralOffice;
};

union LoopTypeOpt switch(boolean) {
  case TRUE: LoopType theValue;
};

struct DcSignatureType {
    KiloOhmsTypeOpt              tipToRingResistance;
    KiloOhmsTypeOpt              tipToGroundResistance;
    VoltsTypeOpt                 tipToGroundVolts;
    KiloOhmsTypeOpt              ringToGroundResistance;
    VoltsTypeOpt                 ringToGroundVolts;
};

union DcSignatureTypeOpt switch(boolean) {
  case TRUE: DcSignatureType theValue;
};

struct FullTestResultsType {
    DcSignatureTypeOpt    dcSignature;
    AcSignatureTypeOpt    acSignature;
    CentralOfficeTypeOpt  centralOffice;
    BalanceTypeOpt        balance;
    LoopTypeOpt           loop;
};

struct QuickTestResultsType {
    DcSignatureTypeOpt    dcSignature;
    AcSignatureTypeOpt    acSignature;
```

```
        BalanceTypeOpt        balance;
        LoopTypeOpt           loop;
    };

    struct CentralOfficeTestResultsType {
        DcSignatureTypeOpt    dcSignature;
        AcSignatureTypeOpt    acSignature;
        CentralOfficeTypeOpt  centralOffice;
        BalanceTypeOpt        balance;
        LoopTypeOpt           loop;
    };

    struct LoopTestResultsType {
        DcSignatureTypeOpt    dcSignature;
        AcSignatureTypeOpt    acSignature;
        CentralOfficeTypeOpt  centralOffice;
        BalanceTypeOpt        balance;
        LoopTypeOpt           loop;
    };

    struct StartToneTestResultsType {
        StringTypeOpt            startToneTestResults;
    };

    struct StopToneTestResultsType {
        StringTypeOpt            stopToneTestResults;
    };

    union PotsTestResultsType switch(PotsTestTypeType) {
case UnknownPotsTestType: NullType                      unknownPotsTest;
       case FullTest:           FullTestResultsType          full;
      case QuickTest:          QuickTestResultsType         quick;
      case CentralOfficeTest:  CentralOfficeTestResultsType centralOffice;
      case LoopTest:           LoopTestResultsType          loop;
       case StartTest:             StartToneTestResultsType  startTone;
       case StopTest:              StopToneTestResultsType   stopTone;
    };

    union PotsTestResultsTypeOpt switch(boolean) {
      case TRUE: PotsTestResultsType theValue;
    };

    struct PotsTestResponseType {
        StringTypeOpt                summary;
        PotsTestResultsTypeOpt       results;
    };


    //-------------------------------------------------------------
    //
    // PotsTestActionPerformer: Provides a service interface for
    //                          testing POTS services (based on
    //                          ANSI T1.262).
    //
    interface PotsTestActionPerformer {


        //
        // testRequestUncontrolled(): initiates a MLT on specified POTS
        //
        void testRequestUncontrolled(
            in  RequestIdType              requestId,
            in  IdentityType               customer,
            in  PotsTestRequestType        request,
            out PotsTestResponseType       response,
```

```
            inout TagValueListTypeOpt       appSpecificData
        )
        raises(
            FallBackReportingError,
            TroubleRepairInProgress,
            ServiceInUse,
            NotFound,
            AccessDenied,
            MissingData,
            InvalidDataReceived,
            ProcessingFailureError
        );


    }; // PotsTestActionPerformer

};

#pragma prefix ""

#endif // ansi_t1_262_IDL
```

Annex C
(informative)

C Supporting Productions from T1.262-1998

This informative annex provides (for historical purposes) the Supporting Productions from T1.262-1998. The revised version is found in clause A.1.12, *Supporting Productions*, of this standard.

```
STF {iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840 10055)  stf(0)}

DEFINITIONS IMPLICIT TAGS ::=  BEGIN

-- EXPORTS Everything

IMPORTS

ObjectInstance
FROM CMIP-1 {joint-iso-ccitt(2) ms(9) cmip(1) modules(0) protocol(3)}

NameType
FROM ASN1DefinedTypesModule  { ccitt recommendation m(13) gnm(3100) informationModel(0)
asn1Modules(2)  asn1DefinedTypesModule(0) };

stfObjectClass OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) objectClass(3)}

stfPackage OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840 10055)
stf(0) package(4)}

stfParameter OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) parameter(5)}

stfNameBinding OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) nameBinding(6)}

stfAttribute OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) attribute(7)}

stfBehaviour OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) behaviour(8)}

stfAction OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840 10055)
stf(0) action(9)}
stfNotification OBJECT IDENTIFIER::={iso(1) member-body(2) usa(840) ansi-t1-262-1998(1 2 840
10055) stf(0) notification(10)}

-- The default value for this syntax is full.
PotsTestRequest ::= INTEGER {
      full               (1),
      quick              (2),
      centralOffice      (3),
      loop               (4),
      ...
}
PotsTestResults ::= SEQUENCE {
      summary            [0] GraphicString,
      results            [1] CHOICE {
            full               [0]   FullTestResults,
            quick              [1]   QuickTestResults,
            centralOffice [2]   CentralOfficeTestResults,
            loop               [3]   LoopTestResults,
```

```
            ...
      } OPTIONAL
}

FullTestResults ::= SEQUENCE {
dcSignature         [1] DcSignature           OPTIONAL,
acSignature         [2] AcSignature           OPTIONAL,
centralOffice       [3] CentralOffice         OPTIONAL,
balance             [4] Balance               OPTIONAL,
loop                [5] Loop                  OPTIONAL,
...
}

QuickTestResults ::= SEQUENCE {
      dcSignature         [1] DcSignature     OPTIONAL,
      acSignature         [2] AcSignature     OPTIONAL,
      balance             [3] Balance         OPTIONAL,
      loop                [4] Loop            OPTIONAL
}

CentralOfficeTestResults ::= SEQUENCE {
      dcSignature         [1] DcSignature           OPTIONAL,
      acSignature         [2] AcSignature           OPTIONAL,
      centralOffice [3] CentralOffice         OPTIONAL,
      balance             [4] Balance               OPTIONAL,
      loop                [5] Loop                  OPTIONAL,
...
}

LoopTestResults ::= SEQUENCE {
      dcSignature         [1] DcSignature           OPTIONAL,
      acSignature         [2] AcSignature           OPTIONAL,
      centralOffice [3] CentralOffice         OPTIONAL,
      balance             [4] Balance               OPTIONAL,
      loop                [5] Loop                  OPTIONAL
}

AcSignature ::= SEQUENCE {
      ringers                   [0] INTEGER         OPTIONAL,
      tipToRingResistance [1] KiloOhmns OPTIONAL,
      tipToGroundResistance     [2] KiloOhmns OPTIONAL,
      ringToGroundResistance    [4] KiloOhmns OPTIONAL
}

Balance ::= SEQUENCE {
      capacitive          [0] Percent,
      longitudinal [1] Decibels
}

CentralOffice ::= SEQUENCE {
      lineCircuitStatus   [0] GraphicString,
      dialToneStatus      [1] GraphicString
}




DcSignature ::= SEQUENCE {
      tipToRingResistance [0] KiloOhmns OPTIONAL,
      tipToGroundResistance     [1] KiloOhmns OPTIONAL,
      tipToGroundVolts          [2] Volts           OPTIONAL,
      ringToGroundResistance    [3] KiloOhmns OPTIONAL,
      ringToGroundVolts         [4] Volts           OPTIONAL
}
```

```
Decibels ::= INTEGER

KiloOhmns ::= INTEGER

Length ::= INTEGER

Loop ::= CHOICE {
       totalLoop                        [0] Length,
       openDistanceFromCentralOffice    [1] Length
}

Percent ::= INTEGER(0..100)

Volts ::= INTEGER

TroubleRepairInProgress ::= GraphicString

END  -- STF
```