TECHNICAL REPORT

TR-050

COBRA v2 for ADSL EMS-NMS Interface

Issue: 1.0

Issue Date: August 2002

Technical comments or questions about this Technical Report should be directed to:

EDITORS:	Andrew J. Mayer, Ph.D.	Telcordia Technologies, Inc.	amayer@telcordia.com
	Rajesh Abbi	Alcatel Telecom, Inc.	Rajesh.Abbi@alcatel.com

TABLE OF CONTENTS

Abstract:

This specifications presents an alignment of the DSL Forum CORBA MIB (TR-041) with the foundation CORBA MIBs specified by ITU-T (X.780, M.3120, Q.821.1, Q.822.1). Alignment of the DSL Forum CORBA MIB will help to promote consistent management interfaces across technologies.

1. INTRODUCTION

This specification provides alignment of the DSL Forum CORBA MIB (TR-041) with the foundation CORBA MIBs specified by ITU-T(X.780, M.3120, Q.821.1, Q.822.1). Alignment of the DSL Forum CORBA MIB will help to promote consistent management interfaces across technologies. Alignment will also allow the opportunity for the DSL Forum to submit an aligned CORBA MIB in support of ITU-T efforts for defining technology specific CORBA MIBs. The ATM Forum Network Management working group has accepted and is pursuing a similar effort.

This document specifies the network management interface between EMS and NMS based on CORBA[1]. The object model is based on TR-035[2], which specifies the Protocol Independent Object Model for ADSL EMS-NMS Interface, and is an evolution of TR-041[10], the CORBA Specification for ADSL EMS-NMS Interface.

2. CORBA BASED NETWORK MANAGEMENT FRAMEWORK

This interface definition uses the CORBA management framework specified by the ITU-T, namely M.3120, Q.821.1, and Q.822.1 as a basis. This document extends that infrastructure to cover CORBA definitions for ADSL related objects specified in TR-035.

NOTE:
This version of the specification mainly covers configuration and performance management related functions for the ADSL EMS-NMS interface. Other functions will be addressed in a future revision.

3. CORBA MANAGEMENT INFORMATION MODEL

The CORBA management information model specified in this document is based on TR-035, which specifies the protocol independent object model to be used at the EMS-NMS interface for managing ADSL access subnetworks. A number of objects in TR-035 are adopted from standard object models specified by the ITU.

Table 3-1 below lists the entities (managed object classes) specified in TR-035 for managing the ADSL access subnetwork, as well as shows the mapping of these to relevant CORBA IDL interface definitions.

This specification is based on the CORBA management framework specified by ITU-T M.3120. ATM specific aspects are modeled by the ATM Forum in its M4 Network View Interface CORBA v2 Specification. [4]
CORBA IDL definitions for ADSL related objects are specified in this document.

Table 3-1. TR-035 to CORBA IDL Mapping Table

TR-035 ENTITY (OBJECT CLASS)	CORBA IDL INTERFACE
Access Subnetwork	M.3120 Network
Layer Network Domain	M.3120 LayerND
Element Management System	-- NA --
Network Element	–M.3120 ManagedElement
Equipment	–M.3120 Equipment
Equipment Holder	–M.3120 EquipmentHolder
Plug-in Unit	–M.3120 CircuitPack
ADSL Line	ADSLLine
ADSL Channel	ADSLChannel
ADSL Configuration Profile	ADSLConfigurationProfile
ATM Subnetwork	[M4] AtmSubnetwork
ATM Subnetwork Connection	[M4] AtmSNC
ATM Link End	[M4] AtmLinkEnd
ATM Network CTP	[M4] AtmNetworkCTP
ATM Network TTP	[M4] AtmNetworkTTP
ATM Network Access Profile	[M4] AtmNetworkAccessProfile
ATM Traffic Descriptor	[M4] AtmTrafficDescABR, AtmTrafficDescCBR, AtmTrafficDescVBR, AtmTrafficDescUBR, AtmTrafficDescGFR

NOTE:

[M4] : These CORBA interfaces are specified by the ATM Forum[4].
-- NA -- : These CORBA interfaces are presently not standardized under the current ATM Forum CORBA framework.

TR-035 only covers configuration management related objects. This specification adds support for additional objects for performance management. Following IDL interfaces for performance management are specified in ITU-T Q.822.1 CORBA framework[4]:

- Scanner
- CurrentData
- HistoryDataIterator
- ThresholdData
- HD File
- AbstractHDScanner
- ListHDScanner
- ClassHDScanner

This specification specifies IDL interface definitions for following additional objects that add performance management support for ADSL objects specified in TR-035. These interfaces are derived from the basic performance management interfaces listed above.

- ADSLLineAtucCurrentData
- ADSLLineAtucCurrentDataFactory
- ADSLLineAturCurrentData
- ADSLLineAturCurrentDataFactory
- ADSLChannelAtucCurrentData
- ADSLChannelAtucCurrentDataFactory
- ADSLChannelAturCurrentData
- ADSLChannelAturCurrentDataFactory

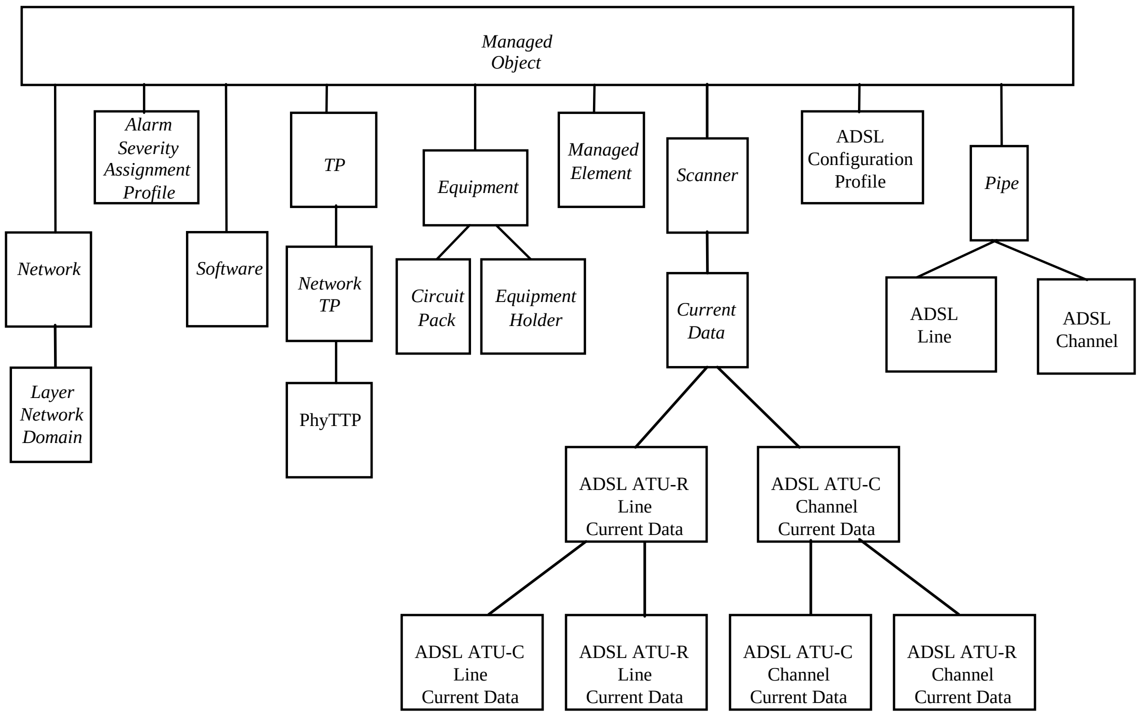
Managed Object
Network
Layer Network Domain
Alarm Severity Assignment Profile
Software
TP
Network TP
PhyTTP
Equipment
Circuit Pack
Equipment Holder
Managed Element
Scanner
Current Data
ADSL ATU-R Line Current Data
ADSL ATU-C Channel Current Data
ADSL ATU-C Line Current Data
ADSL ATU-R Line Current Data
ADSL ATU-C Channel Current Data
ADSL ATU-R Channel Current Data
ADSL Configuration Profile
Pipe
ADSL Line
ADSL Channel

Figure 3-1 Inheritance Diagram

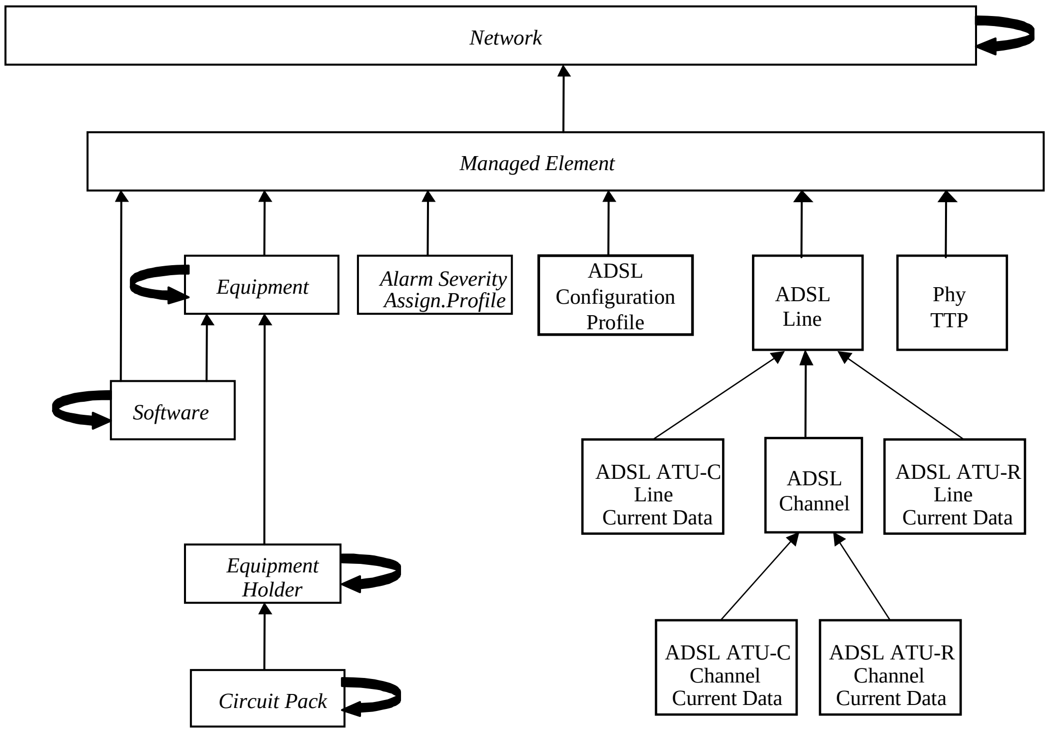
Network
Managed Element
Equipment
Alarm Severity Assign.Profile
ADSL Configuration Profile
ADSL Line
Phy TTP
Software
ADSL ATU-C Line Current Data
ADSL Channel
ADSL ATU-R Line Current Data
Equipment Holder
ADSL ATU-C Channel Current Data
ADSL ATU-R Channel Current Data
Circuit Pack

Figure 3-2 Containment Diagram

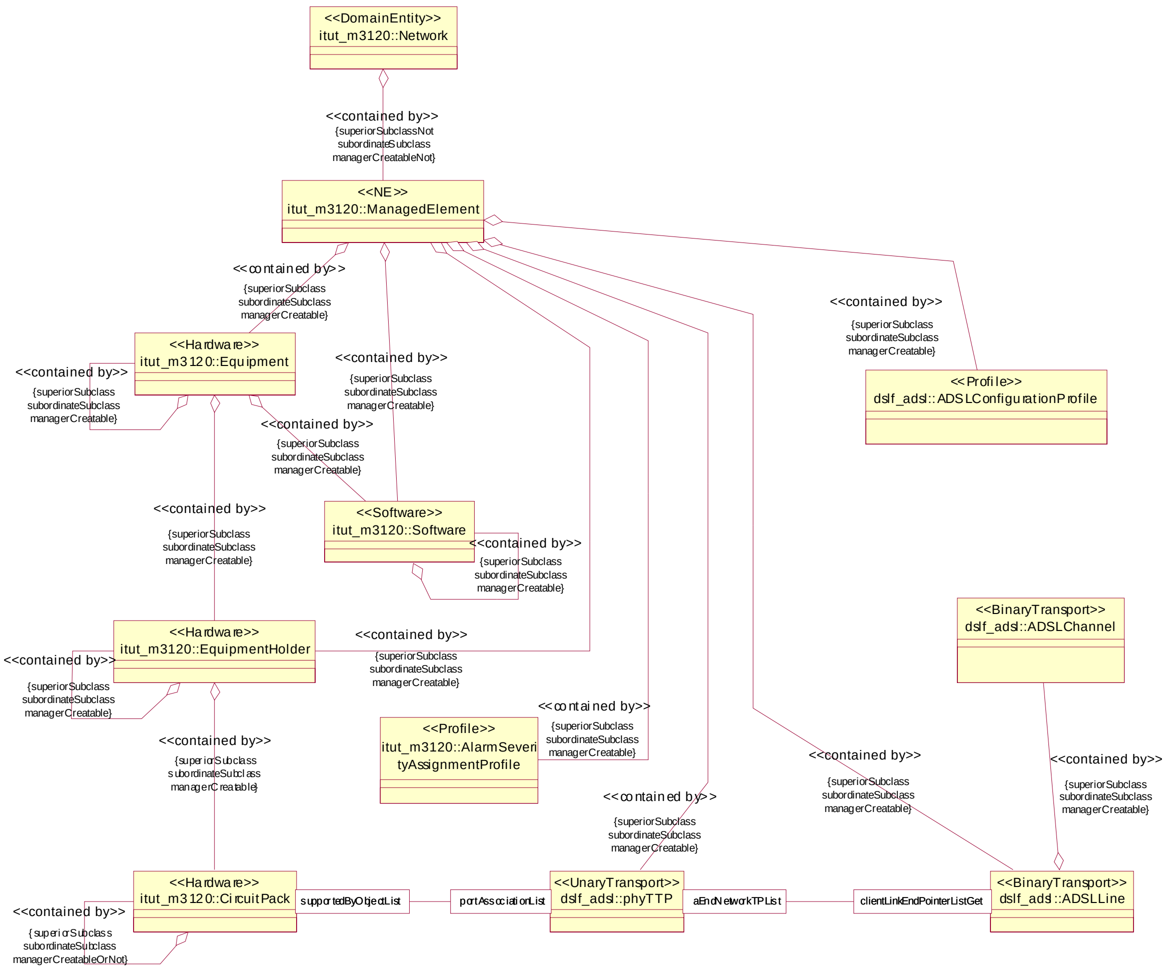

Figure 3-3 UML Object Relationship Diagram

REFERENCES

[1] The Object Management Group (OMG), “The Common Object Request Broker: Architecture and Specification”, OMG Document: formal/99-10-07, Revision 2.3.1, October 1999.

[2] DSL Forum (DSLF), “Protocol Independent Object Model for ADSL EMS-NMS Interface.”, DSLF Technical Report: TR-035, March 2000.

[3] The ATM Forum (ATMF), “M4 Interface Requirements and Logical MIB: ATM Network View”, version 2, ATMF Specification: af-nm-0058.001, May 1999.

[4] The ATM Forum (ATMF), “CORBA Specification version 2 for M4 Interface: Network View”, ATMF Final Ballot Document: FB-NM-0185.000, May 2002.

[5] DSL Forum (DSLF), “Proposed IDL definitions for Performance Management portion of WT-046v1”, Alcatel (R. Abbi & A. Tuzel), DSLF Contribution : dslforum2000.113, May 2000.

[6] DSL Forum (DSLF), “Proposed updates to the Performance Management CORBA IDL definitions”, Telcordia Technologies (A. Mayer, B. Atwater, K. Armington), DSLF Contribution: dslforum2000.258, August 2000.

[7] DSL Forum (DSLF), “Update of WT-046v3 IDL definitions for Performance Management”, Alcatel (R. Abbi & A. Tuzel) and Telcordia Technologies (A. Mayer & K. Armington), DSLF Contribution: dslforum2000.428, December 2000.

[8] ITU-T, SG4, RECOMMENDATION M.3120, “CORBA GENERIC NETWORK AND NE LEVEL INFORMATION MODEL”, October 2001.

[9] ITU-T, SG4, RECOMMENDATION Q.822.1, “CORBA-based TMN Performance Management Service”, October 2001.

[10] DSL Forum (DSLF), “CORBA Specification for ADSL EMS-NMS Interface.”, DSLF Technical Report: TR-041, June 2001.

/**

Appendix A : IDL For ADSL Management Version 2

DSL Forum_v2 CORBA idl model aligned with ITU-T M.3120, Q.822.1
Prepared by Andrew J. Mayer, Ph.D.
Telcordia Technologies
Email: amayer@telcordia.com
*/

/**
GENERAL COMMENTS

ID attributes for all DSL Forum objects have been removed, since they need not be defined explicitly in the M.3120 model. All other reference to IDs in operations is replaced by the X.780:MONameType structure which has the object ID as an embedded attribute.

Other attributes relevant to the DSL information model that are inherited from superior M.3120 model have been kept in this idl code, commented out, for reference purposes. The M.3120 corresponding attribute is also pointed out in the comment.

*/

/**

A.1 : ADSL Configuration Management Module (dslf_adsl_v2.idl)

*/

```
#ifndef _dslf_adsl_v2_idl_
#define _dslf_adsl_v2_idl_

#include <itut_x780.idl>
#include <itut_x780ct.idl>
#include <itut_m3120.idl>
#include <itut_x780_1.idl>
#include <itut_m3120const.idl>
#include <itut_q822_1.idl>

#pragma prefix "dslforum.org"


/**
This IDL code is intended to be stored in a file named "dslf_adsl_v2.idl"
located in the search path of your IDL compiler.

All performance management aspects are grouped under a separate sub-module,
dslf_adsl_pm_v2 to facilitate extensions of this IDL.
*/


/*****************************************************************************/
```

```
/*  This module defines the interfaces for managing ADSL transmission lines. */
/*  It is based on the CORBA Network Management Framework defined in the     */
/*  Interfaces defined in this module are based on the ADSL Management       */
/*  Object Model specified in TR-035.                                        */
/*************************************************************************/
module dslf_adsl_v2 {

const string moduleName = “dslf_adsl_v2”;


/**
```

A.1.1 : Imports

```
*/

/**
```

A.1.1.1 Types imported from itut_x780

```
*/

      typedef itut_x780::AdministrativeStateType AdministrativeStateType;
      typedef itut_x780::BooleanTypeOpt BooleanTypeOpt;
      typedef itut_x780::DeletePolicyType DeletePolicyType;
      typedef itut_x780::Istring Istring;
      typedef itut_x780::NameBindingType NameBindingType;
      typedef itut_x780::MONameType MONameType;
      typedef itut_x780::MONameSetType MONameSetType;
      typedef itut_x780::ObjectClassType ObjectClassType;
      typedef itut_x780::ObjectClassSetType ObjectClassSetType;
      typedef itut_x780::OperationalStateType OperationalStateType;
      typedef itut_x780::StringSetType StringSetType;


/**
```

A.1.1.2 Interfaces imported from itut_m.3120

```
typedef itut_m3120::Network_F Network_F;
*/

/**
```

A.1.1.3 Valuetypes imported from itut_m.3120

```
*/

typedef itut_m3120::NetworkValueType NetworkValueType;
typedef itut_m3120:: PointDirectionalityType PointDirectionalityType;

/**
```

A.1.1.4 Interfaces imported from itut_x.780

```
typedef itut_x780::ManagedObject_F ManagedObject_F;
typedef itut_x780::ManagedObjectFactory ManagedObjectFactory;
*/
```

```
/**
```

A.1.1.5 Valuetypes imported from itut_x.780

```
*/
typedef itut_x780::ManagedObjectValueType ManagedObjectValueType;


/**
```

A.1.1.6 Exceptions imported from itut_x.780

```
itut_x780::ApplicationError
itut_x780::CreateError
itut_x780::DeleteError

These exceptions all produce an attribute that gives extra details pertaining
to specific error conditions. A few predefined attribute values are listed in
X.780. These values can be used to replace dslf_adsl_v2 exceptions with X.780
exceptions.

Throughout this document, dslf exceptions were mapped to the following x.780
exceptions:

NetMgmt::ObjectsFailure is mapped to x.780::ApplicationError with no specific
error condition.

NetMgmt::DuplicateName is mapped to x.780::CreateError with duplicateName as a
specific error condition.

NetMgmt::NotSupported is mapped to x.780::CreateError with unsupportedPackages
as a specific error condition.

NetMgmt::ContainedObjects is mapped to x.780::DeleteError with containsObjects
as a specific error condition.

NetMgmt::DeleteNotAllowed is mapped to x.780::DeleteError with notDeletable as
a specific error condition.

*/
```

```
/**
```

A.1.2 Typedefs forward declarations

```
*/


/** Reference to an ADSL Configuration Profile */
typedef MONameType ADSLConfigurationProfileNameType;


/** Reference to an ADSL Channel */
typedef MONameType ADSLChannelNameType;


/** Reference to an ADSL Line */
typedef MONameType ADSLLineNameType;


/**
```

A.1.3 Structures and Typedefs

```
*/



/** Options for ADSL line codes */
enum LineCodingType {
    lineCoding_dmt,
    lineCoding_cap,
    lineCoding_qam,
    lineCoding_other
};

/** Options for ADSL Channel latency */
enum ChannelType {
    channel_noChannel,
    channel_fastOnly,
    channel_interleavedOnly,
    channel_fastAndInterleaved,
    channel_fastOrInterleaved
};

/** Options for ADSL line rate mode */
enum RateModeType {
    rateMode_fixedRate,
    rateMode_adaptAtStartup,
    rateMode_adaptAtRuntime
};

/** Options for modem operating modes */
enum OperationalMode {
    operationalMode_ansi,
    operationalMode_etsi,
    operationalMode_potsNonOverlapped,
    operationalMode_potsOverlapped,
    operationalMode_isdnNonOverlapped,
    operationalMode_isdnOverlapped,
    operationalMode_isdnTcm,
    operationalMode_potsNonOverlappedLite,
    operationalMode_potsOverlappedLite,
    operationalMode_isdnTcmLite
};

/** List of Operational Modes */
typedef sequence <OperationalMode> OperationalModeSetType;

/** Options for ADSL line conditions */
enum LineCondition {
    lineCondition_atuc_lof,
    lineCondition_atuc_los,
    lineCondition_atuc_lpr,
    lineCondition_atuc_lol,
    lineCondition_atuc_lossOfSignalQuality,
    lineCondition_atuc_dataInitfailure,
    lineCondition_atuc_configInitFailure,
    lineCondition_atuc_protocolInitFailure,
    lineCondition_atuc_noPeerPresent,
    lineCondition_atuc_bitRateThreshold,
    lineCondition_atur_lof,
    lineCondition_atur_los,
    lineCondition_atur_lpr,
    lineCondition_atur_lossOfSignalQuality,
    lineCondition_atur_bitRateThreshold
};

/** List of ADSL line conditions */
typedef sequence <LineCondition> LineConditionSetType;

/** ADSL Line related operational data */
struct LineData {

    /** This read-only attribute provides the Line Coding information */
    LineCodingType lineCoding;

    /** This read-only attribute provides the Current signal/noise margin
       for the received signal in 1/10th of a dB.*/
    long currentSNRMargin;

    /** This read-only attribute indicates the measured difference in
       total power transmitted by the peer ATU and the total power
       received by this ATU in 1/10th of a dB */
    long currentAttenuation;

    /** This read-only attribute indicates the measured power transmitted
       by the ATU in 1/10th of a dB */
    long currentOutputPower;

    /** This read-only attribute indicates the maximum attainable rate
       in kbps */
    long currentAttainableRate;

    /** This read-only attribute indicates the current line rate in kbps */
    long currentLineRate;

    /** This read-only attribute indicates the previous line rate in kbps */
    long previousLineRate;
    /** This attribute indicates the supported Channel types over this
       ADSL line */
    ChannelType supportedChannelTypes;

};




/** ADSL Channel operating data */
struct ChannelData {

    /** This read-only attribute indicates the current Channel rate in kbps */
    long currentChannelRate;

    /** This read-only attribute indicates the previous Channel rate in kbps */
    long previousChannelRate;

    /**This attribute indicates the current interleave delay in mili-seconds */
    long interleaveDelay;  /* For Interleaved Channel Only */

    /** This attribute indicates the current length, in bytes, of the
       Channel data-block on which CRC is calculated */
    long crcBlockLength;
};
```

```
/** ADSL Line configuration parameters */
struct LineConf {

    /** This attribute configures the modem rate adaptation mode */
    RateModeType rateMode;

    /** This attribute configures the target signal/noise margin, in 1/10th
       of dB, the modem must achieve with BER of 10-7 or better */
    long targetSNRMargin;

    /** This attribute configures the maximum signal/noise margin, in 1/10th
       of dB, the modem must try to maintain before increasing the data rate */
    long maxSNRMargin;

    /** This attribute configures the minimum acceptable signal/noise margin,
       in 1/10th of dB */
    long minSNRMargin;
};

/** ADSL Channel configuration parameters */
struct ChannelConf {

    /** This attribute configures the minimum acceptable transmission rate,
       for the Channel in bps */
    long minTxRate;

    /** This attribute configures the maximum allowed transmission rate,
       for the Channel in bps */
    long maxTxRate;

    /** This attribute configures the maximum allowed interleave delay,
       in miliseconds, for the interleaved Channel */
    long maxInterleaveDelay; /* For Interleaved Channel Only */
};

/**
Port ID, managed element and port required, others optional
*/
      struct PortIDType
      {
            Istring     managedElement;
            Istring     bay;
            Istring     shelf;
            Istring     drawer;
            Istring     slot;
            Istring     port;
      };


/**
```

A.1.4 Valuetypes, Interfaces and Factories - Facade

```
*/

/**
```

A.1.4.1 ADSL Line_F

```
*----------------------------------------------------------------
 *         ADSL Line Interface
 *----------------------------------------------------------------
 * This interface represents an ADSL Line.
 * Instances of this object may creation by factory or automatically
 * by managed system.
 *----------------------------------------------------------------
 */

valuetype ADSLLineValueType: itut_m3120::PipeValueType
{

/**   ADSLLineValueType uses the following inherited from
itut_m3120:: AbstractPipeValueType:

//    to provide the relationship to the physical port (PhyTTP)
// use m3120::Pipe      public MONameSetType aEndNetworkTPList;
                  // GET, SET-BY-CREATE


//    to represent underlyingEquipmentUnits:
// use m3120::Pipe      public MONameSetType          supportedByObjectList;
                  // GET-REPLACE, ADD-REMOVE


//    to represent adminState:
// use m3120::Pipe      public AdministrativeStateType
      administrativeState;
                  // GET-REPLACE


//    to represent operState:
// use m3120::Pipe      public OperationalStateType   operationalState;
                  // GET


//    to represent userLabel;
// use m3120::Pipe public Istring               userLabel;
                  // GET-REPLACE
**/

    public BooleanTypeOpt                     initFailureSwitch;
                  // GET-REPLACE

    public LineData                   lineAtucData;
                  // GET

    public LineData                   lineAturData;
                  // GET
    public LineConditionSetType             lineStatus;
                  // GET

    public ADSLConfigurationProfileNameType configurationProfile;
                  // GET-REPLACE


    public OperationalMode            currentOperationalMode;
                  // GET

    public OperationalModeSetType           allowedOperationalModes;
                  // GET, SET-BY-CREATE

    public OperationalModeSetType           supportedOperationalModes;
                  // GET-REPLACE

}; // valuetype ADSLLineValueType

interface ADSLLine_F : itut_m3120::Pipe_F {

/**     ADSLLine_F inherits the following methods from
        itut_m3120:: Pipe_F:
        supportedByObjectListGet, administrativeStateGet, administrativeStateSet,
        operationalStateGet, userLabelGet, aEndNetworkTPListGet

**/


    /** Get the equipment unit(s) supporting an ADSL Line, supported in
itut_m3120:: Pipe_F */
//          MONameSetType supportedByObjectListGet
//                (in MONameType name)
//                raises (itut_x780::ApplicationError,
//                      NOsupportedByPackage);



    /** Configure a Channel for an ADSL Line */
    void createChannel     (in MONameType name,
                            in  ChannelType   chanType,
                            out ADSLChannelNameType channelId)
        raises (itut_x780::ApplicationError, itut_x780::CreateError);

    /** Get the Administrative State of an ADSL Line, supported in itut_m3120::
Pipe_F */
//          AdministrativeStateType administrativeStateGet
//                (in MONameType name)
//                raises (itut_x780::ApplicationError,
//                      NOadministrativeStatePackage);

    /** Set the Administrative State of an ADSL Line, supported in itut_m3120::
Pipe_F */
//          void administrativeStateSet
//                (in MONameType name,
//                in AdministrativeStateType administrativeState)
//                raises (itut_x780::ApplicationError,
//                      NOadministrativeStatePackage);

    /** Get the Operational State of an ADSL Line, supported in itut_m3120::
Pipe_F */
```

```
//          OperationalStateType operationalStateGet
//                (in MONameType name)
//                raises (itut_x780::ApplicationError,
//                      NOoperationalStatePackage);

/** Get the User Label of an ADSL Line, supported in itut_m3120:: Pipe_F */
//          Istring userLabelGet
//                (in MONameType name)
//                raises (itut_x780::ApplicationError,
//                      NOuserLabelPackage);

/** Set the User Label of an ADSL Line, supported in itut_m3120:: Pipe_F */
//          void userLabelSet
//                (in MONameType name,
//                in Istring userLabel)
//                raises (itut_x780::ApplicationError,
//                      NOuserLabelPackage);


    /** Get modem initialization failure notification flag.
        Initialization failure alarms are reported if 'True' */
    BooleanTypeOpt initFailureSwitchGet (in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Set modem initialization failure notification flag.
        Initialization failure alarms are reported if 'True' */
    void initFailureSwitchSet (in MONameType name,
            in BooleanTypeOpt initFailureSwitch)
        raises (itut_x780::ApplicationError);


    /** Get operating data for the ATU-C end of an ADSL Line */
    LineData lineAtucDataGet (in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Get operating data for the ATU-R end of an ADSL Line */
    LineData lineAturDataGet (in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Get the Status of an ADSL Line */
    LineConditionSetType lineStatusGet (in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Get the ADSL Configuration Profile associated with an ADSL Line */
    ADSLConfigurationProfileNameType aDSLConfigurationProfileGet
                                (in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Assign a new Configuration Profile to an ADSL Line */
    void aDSLConfigurationProfileSet(in MONameType name,
                                  in ADSLConfigurationProfileNameType
confProfile )
        raises (itut_x780::ApplicationError);


    /** Get current modem operational mode */
    OperationalMode currentOperationalModeGet(in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Get operational modes supported by an ADSL Line */
    OperationalModeSetType supportedOperationalModesGet(in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Get operational modes allowed for an ADSL Line */
    OperationalModeSetType allowedOperationalModesGet(in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Set operational modes allowed for an ADSL Line */
    void allowedOperationalModesSet(in MONameType name,
                             in OperationalModeSetType allowedModes)
        raises (itut_x780::ApplicationError);

            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectCreation)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectDeletion)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, stateChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, attributeValueChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, communicationAlarm)



}; // end of ADSLLine interface

/**------------------------------------------------------------------
 *         ADSL Line Factory Interface
 *-----------------------------------------------------------------
 * This interface is used to create new instances of ADSL Line.
 *-----------------------------------------------------------------
 */
interface ADSLLineFactory : itut_x780::ManagedObjectFactory  {



    /** Create a new instance of an ADSL Line */
       itut_x780::ManagedObject create
               (in NameBindingType nameBinding,
                       // module name containing Name Binding info.
               in MONameType superior,
                       // Name of containing object.
               in string reqID,
                       // Requested ID value for name, will be
                       // empty if auto-naming is to be used.
               out MONameType name,
                       // Entire name of newly created object.
               in StringSetType packageNameList,
                       // List of packages requested.

               in MONameSetType              aEndNetworkTPList,
               in MONameSetType              supportedByObjectList,
               in AdministrativeStateType    administrativeState,
               in Istring                    userLabel,
               in BooleanTypeOpt             initFailureSwitch,
               in ADSLConfigurationProfileNameType configurationProfile,
               in OperationalModeSetType     allowedOperationalModes,
               in OperationalModeSetType     supportedOperationalModes)

        raises (itut_x780::ApplicationError,
                        itut_x780::CreateError);
```

```
}; // end of ADSLLineFactory


/**
```

A.1.4.2 ADSL Channel_F

```
*----------------------------------------------------------------
 *          ADSL Channel Interface
 *----------------------------------------------------------------
 * This interface represents an ADSL Channel. Both ends of the channel
 * (ATU-C and ATU-R) are represented by this interface.
 * Instances of ADSLChannel are created using the createChannel method
 * of the ADSLLine object, or automatically by managed system.
 *----------------------------------------------------------------
 */
valuetype ADSLChannelValueType: itut_m3120::PipeValueType
{

/**   ADSLChannelValueType uses the following inherited from
itut_m3120:: AbstractPipeValueType:


//    to represent adminState:
// use m3120::Pipe      public AdministrativeStateType
      administrativeState;
                  // GET-REPLACE


//    to represent operState:
// use m3120::Pipe      public OperationalStateType   operationalState;
                  // GET

    public ChannelType        channelTypeValue;
                  // GET, SET-BY-CREATE

    public ChannelData        atucData;
                  // GET

    public ChannelData        aturData;
                  // GET

}; // valuetype ADSLChannelValueType


interface ADSLChannel_F : itut_m3120::Pipe_F {


/**   ADSLChannel_F inherits the following methods from
      itut_m3120:: Pipe_F:
      administrativeStateGet, administrativeStateSet, operationalStateGet

**/



    /** Get the Administrative State of an ADSL Channel, supported in
itut_m3120::Pipe_F */
//          AdministrativeStateType administrativeStateGet
//                (in MONameType name)
//                raises (itut_x780::ApplicationError,
//                      NOadministrativeStatePackage);



    /** Set the Administrative State of an ADSL Channel, supported in
itut_m3120::Pipe_F */
//          void administrativeStateSet
//                (in MONameType name,
//                in AdministrativeStateType administrativeState)
//                raises (itut_x780::ApplicationError,
//                      NOadministrativeStatePackage);

    /** Get the Operational State of an ADSL Channel, supported in
itut_m3120::Pipe_F */
//          OperationalStateType operationalStateGet
//                (in MONameType name)
//                raises (itut_x780::ApplicationError,
//                      NOoperationalStatePackage);

    /** Get the Channel type (latency) of an ADSL Channel */
    ChannelType channelTypeGet(in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Get data for the ATU-C end of an ADSL Channel */
    ChannelData channelAtucDataGet(in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Get data for the ATU-R end of an ADSL Channel */
    ChannelData channelAturDataGet(in MONameType name)
        raises (itut_x780::ApplicationError);
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectCreation)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectDeletion)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, stateChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, attributeValueChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, communicationAlarm)


}; // end of ADSLChannel interface

/**
```

A.1.4.3 ADSL Configuration Profile_F

```
*-----------------------------------------------------------------
 *        ADSL Configuration Profile Interface
 *-----------------------------------------------------------------
 * This interface represents an ADSL Configuration Profile.
 * Configuration Profiles are created via the
 * ADSLConfigurationProfileFactory Interface.
 *-----------------------------------------------------------------
 */
```

```
valuetype ADSLConfigurationProfileValueType: itut_x780::ManagedObjectValueType
{
    public Istring            profileName;
                  // GET, SET-BY-CREATE
    public LineConf           lineConfAtuc;
                  // GET, SET-BY-CREATE
    public LineConf           lineConfAtur;
                  // GET, SET-BY-CREATE
    public ChannelConf        fastChannelConfAtuc;
                  // GET, SET-BY-CREATE
    public ChannelConf        fastChannelConfAtur;
                  // GET, SET-BY-CREATE
    public ChannelConf        interleavedChannelConfAtuc;
                  // GET, SET-BY-CREATE
    public ChannelConf        interleavedChannelConfAtur;
                  // GET, SET-BY-CREATE

}; // valuetype ADSLChannelValueType


interface ADSLConfigurationProfile_F : itut_x780::ManagedObject_F {


    /** Get ATU-C configuration data associated with a Configuration Profile */
    LineConf lineConfAtucGet(in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Get ATU-R configuration data associated with a Configuration Profile */
    LineConf lineConfAturGet(in MONameType name)
        raises (itut_x780::ApplicationError);


    /** Get ATU-C Fast Channel configuration data associated with a
        Configuration Profile */
    ChannelConf fastChannelConfAtucGet(in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Get ATU-R Fast Channel configuration data associated with a
        Configuration Profile */
    ChannelConf fastChannelConfAturGet(in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Get ATU-C Interleaved Channel configuration data associated with a
        Configuration Profile */
    ChannelConf interleavedChannelConfAtucGet(
                                    in MONameType name)
        raises (itut_x780::ApplicationError);

    /** Get ATU-R Interleaved Channel configuration data associated with a
        Configuration Profile */
    ChannelConf interleavedChannelConfAturGet(
                                    in MONameType name)
        raises (itut_x780::ApplicationError);

            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectCreation)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectDeletion)
}; // end of ADSLConfigurationProfile interface


/**------------------------------------------------------------------
 *         ADSL Configuration Profile Factory Interface
 *------------------------------------------------------------------
 * This interface is used to create new instances of ADSL Configuration
 * Profiles.
 *------------------------------------------------------------------
 */
interface ADSLConfigurationProfileFactory : itut_x780::ManagedObjectFactory  {



    /** Create a new instance of an ADSL Configuration Profile */
       itut_x780::ManagedObject create
               (in NameBindingType nameBinding,
                       // module name containing Name Binding info.
               in MONameType superior,
                       // Name of containing object.
               in string reqID,
                       // Requested ID value for name, will be
                       // empty if auto-naming is to be used.
               out MONameType name,
                       // Entire name of newly created object.
               in StringSetType packageNameList,
                       // List of packages requested.


                        in Istring        profileName,
                        in LineConf      lineConfAtuc,
                        in LineConf      lineConfAtur,
                        in ChannelConf   fastChannelConfAtuc,
                        in ChannelConf   fastChannelConfAtur,
                        in ChannelConf   interleavedChannelConfAtuc,
                        in ChannelConf   interleavedChannelConfAtur)
        raises (itut_x780::ApplicationError,
                        itut_x780::CreateError);

}; // end of ADSLConfigurationProfileFactory


/**
```

A.1.4.4 PhyTTP_F

```
*/
      /*    For ADSL the PhyTTP may optionally be used to represent physical
      port inventory.  If implemented, this object should minimally provide the
      portId and a pointer to the supported ADSLLine (clientLinkEndPointerList)
      and the supporting CircuitPack (supportedByObjectList).
      It is expected that this object will be deprecated once ITU-T M.3120
      defines a CORBA object representing the port.
      */

            valuetype PhyTTPValueType: itut_m3120:: NetworkTPValueType
            {
      /**   PhyTTPValueType uses the following inherited from
            itut_m3120:: TPValueType:
                  public MONameSetType                supportedByObjectList;
                  // points to the supporting circuit pack
                  // GET
            public OperationalStateType         operationalState;
                  // conditional, present if an instance supports it.
                  // GET
            public AlarmStatusType              alarmStatus;
                  // conditional, present if the TP supports
            communications
                  // alarm notification.
                  // GET
            public CurrentProblemSetType        currentProblemList;
                  // conditional, present if the TP supports
communications
                  // alarm notification.
                  // GET
            public MONameType
                  alarmSeverityAssignmentProfilePointer;
                  // conditional, present if an instance supports
                  // configuration of alarm severities.
                  // GET-REPLACE


      PhyTTPValueType uses the following inherited from
      itut_m3120:: NetworkTPValueType:
            public PointDirectionalityType      pointDirectionality;
                  // GET
            public SignalIdType           signalId;
                  // GET, SET-BY-CREATE


**/

            public      PortIDType phyTTPPortID;
                  // conditional
                  // present if the server TTP port is represented
                  // GET


            public MONameSetType clientLinkEndPointerList;
                  // GET-REPLACE

      }; // valuetype PhyTTPValueType


      interface PhyTTP_F: itut_m3120::NetworkTP_F
      {
/**   Instances of AtmLinkEndPhy are created using the
AtmLinkEndPhyFactory
      or automatically by the managed system.
**/

/**   PhyTTP_F inherits the following methods from
      itut_m3120:: TP_F:
      operationalStateGet, alarmStatusGet, currentProblemListGet,
      alarmSeverityAssignmentProfilePointerGet,
      alarmSeverityAssignmentProfilePointerSet,
      supportedByObjectListGet

      PhyTTP_F inherits the following methods from
      itut_m3120:: NetworkTP_F:
      pointDirectionalityGet, signalIdGet


**/

            PortIDType phyTTPPortIDGet
                  (in MONameType name)
                  raises (itut_x780::ApplicationError,
                        itut_x780::CreateError);


            MONameSetType clientLinkEndPointerListGet
                  (in MONameType name)
                  raises (itut_x780::ApplicationError);
            void clientLinkEndPointerListSet
                  (in MONameType name,
                  in MONameSetType clientLinkEndPointerList)
                  raises (itut_x780::ApplicationError);


            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectCreation)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectDeletion)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, attributeValueChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, stateChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, communicationAlarm)



      }; // interface PhyTTP_F

      interface PhyTTPFactory:
            itut_x780::ManagedObjectFactory
      {
            itut_x780::ManagedObject create
                  (in NameBindingType nameBinding,
                        // module name containing Name Binding info.
                  in MONameType superior,
                        // Name of containing object.
                  in string reqID,
                        // Requested ID value for name, will be
                        // empty if auto-naming is to be used.
                  out MONameType name,
                        // Entire name of newly created object.
                  in StringSetType packageNameList,
                        // List of packages requested.

                  in MONameType
                  alarmSeverityAssignmentProfilePointer,
                  in PointDirectionalityType    pointDirectionality,

                  in MONameSetType clientLinkEndPointerList)

                  raises (itut_x780::ApplicationError,
                        itut_x780::CreateError);
      }; // interface PhyTTPFactory
```

```
/**
```

A.1.5 Interfaces and Factories – Fine Grain

The behaviour of the Fine-Grained interfaces are identical to the corresponding Façade grained interfaces. Therefore, comments are not included in the fine-grained interfaces. Readers are referred to the Façade interface in Section A.1.4 for the behaviour of the fine-grained interface.

This section can be omitted from IDL if a management system only supports Façade grained interface.

```
*/

/**
```

A.1.5.1 ADSL Line

```
**/

interface ADSLLine : itut_m3120::Pipe {

/**   ADSLLine inherits the following methods from
      itut_m3120:: Pipe:
      supportedByObjectListGet, administrativeStateGet, administrativeStateSet,
      operationalStateGet, userLabelGet, aEndNetworkTPListGet

**/


    /** Get the equipment unit(s) supporting an ADSL Line, supported in
itut_m3120:: Pipe */
//          MONameSetType supportedByObjectListGet ()
//                raises (itut_x780::ApplicationError,
//                      NOsupportedByPackage);



    /** Configure a Channel for an ADSL Line */
    void createChannel     (in  ChannelType   chanType,
                            out ADSLChannelNameType channelId)
        raises (itut_x780::ApplicationError, itut_x780::CreateError);

    /** Get the Administrative State of an ADSL Line, supported in itut_m3120::
Pipe */
//          AdministrativeStateType administrativeStateGet ()
//                raises (itut_x780::ApplicationError,
//                      NOadministrativeStatePackage);

    /** Set the Administrative State of an ADSL Line, supported in itut_m3120::
Pipe */
//          void administrativeStateSet
//                (in AdministrativeStateType administrativeState)
//                raises (itut_x780::ApplicationError,
//                      NOadministrativeStatePackage);
```

```
    /** Get the Operational State of an ADSL Line, supported in itut_m3120::
Pipe */
//          OperationalStateType operationalStateGet ()
//                raises (itut_x780::ApplicationError,
//                      NOoperationalStatePackage);

/** Get the User Label of an ADSL Line, supported in itut_m3120:: Pipe */
//          Istring userLabelGet ()
//                raises (itut_x780::ApplicationError,
//                      NOuserLabelPackage);

/** Set the User Label of an ADSL Line, supported in itut_m3120:: Pipe */
//          void userLabelSet
//                (in Istring userLabel)
//                raises (itut_x780::ApplicationError,
//                      NOuserLabelPackage);


    /** Get modem initialization failure notification flag.
        Initialization failure alarms are reported if 'True' */
    BooleanTypeOpt initFailureSwitchGet ()
        raises (itut_x780::ApplicationError);

    /** Set modem initialization failure notification flag.
        Initialization failure alarms are reported if 'True' */
    void initFailureSwitchSet (in BooleanTypeOpt initFailureSwitch)
        raises (itut_x780::ApplicationError);


    /** Get operating data for the ATU-C end of an ADSL Line */
    LineData lineAtucDataGet ()
        raises (itut_x780::ApplicationError);

    /** Get operating data for the ATU-R end of an ADSL Line */
    LineData lineAturDataGet ()
        raises (itut_x780::ApplicationError);

    /** Get the Status of an ADSL Line */
    LineConditionSetType lineStatusGet ()
        raises (itut_x780::ApplicationError);

    /** Get the ADSL Configuration Profile associated with an ADSL Line */
    ADSLConfigurationProfileNameType aDSLConfigurationProfileGet
                                    ()
        raises (itut_x780::ApplicationError);

    /** Assign a new Configuration Profile to an ADSL Line */
    void aDSLConfigurationProfileSet
            (in ADSLConfigurationProfileNameType confProfile )
        raises (itut_x780::ApplicationError);


    /** Get current modem operational mode */
    OperationalMode currentOperationalModeGet()
        raises (itut_x780::ApplicationError);

    /** Get operational modes supported by an ADSL Line */
    OperationalModeSetType supportedOperationalModesGet()
        raises (itut_x780::ApplicationError);
    /** Get operational modes allowed for an ADSL Line */
    OperationalModeSetType allowedOperationalModesGet()
        raises (itut_x780::ApplicationError);

    /** Set operational modes allowed for an ADSL Line */
    void allowedOperationalModesSet(
                        in OperationalModeSetType allowedModes)
        raises (itut_x780::ApplicationError);

            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectCreation)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectDeletion)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, stateChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, attributeValueChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, communicationAlarm)


}; // end of ADSLLine interface


/**
```

A.1.5.2 ADSL Channel

```
*/


interface ADSLChannel : itut_m3120::Pipe {


/**   ADSLChannel inherits the following methods from
      itut_m3120:: Pipe:
      administrativeStateGet, administrativeStateSet, operationalStateGet

**/



    /** Get the Administrative State of an ADSL Channel, supported in
itut_m3120::Pipe */
//          AdministrativeStateType administrativeStateGet ()
//                raises (itut_x780::ApplicationError,
//                      NOadministrativeStatePackage);



    /** Set the Administrative State of an ADSL Channel, supported in
itut_m3120::Pipe */
//          void administrativeStateSet
//                (in AdministrativeStateType administrativeState)
//                raises (itut_x780::ApplicationError,
//                      NOadministrativeStatePackage);
```

```
    /** Get the Operational State of an ADSL Channel, supported in
itut_m3120::Pipe */
//          OperationalStateType operationalStateGet ()
//                raises (itut_x780::ApplicationError,
//                      NOoperationalStatePackage);

    /** Get the Channel type (latency) of an ADSL Channel */
    ChannelType channelTypeGet()
        raises (itut_x780::ApplicationError);

    /** Get data for the ATU-C end of an ADSL Channel */
    ChannelData channelAtucDataGet()
        raises (itut_x780::ApplicationError);

    /** Get data for the ATU-R end of an ADSL Channel */
    ChannelData channelAturDataGet()
        raises (itut_x780::ApplicationError);

            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectCreation)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectDeletion)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, stateChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, attributeValueChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, communicationAlarm)


}; // end of ADSLChannel interface

/**
```

A.1.5.3 ADSL Configuration Profile

```
*/


interface ADSLConfigurationProfile : itut_x780::ManagedObject {


    /** Get ATU-C configuration data associated with a Configuration Profile */
    LineConf lineConfAtucGet()
        raises (itut_x780::ApplicationError);

    /** Get ATU-R configuration data associated with a Configuration Profile */
    LineConf lineConfAturGet()
        raises (itut_x780::ApplicationError);


    /** Get ATU-C Fast Channel configuration data associated with a
        Configuration Profile */
    ChannelConf fastChannelConfAtucGet()
        raises (itut_x780::ApplicationError);

    /** Get ATU-R Fast Channel configuration data associated with a
        Configuration Profile */
    ChannelConf fastChannelConfAturGet()
        raises (itut_x780::ApplicationError);
    /** Get ATU-C Interleaved Channel configuration data associated with a
        Configuration Profile */
    ChannelConf interleavedChannelConfAtucGet()
        raises (itut_x780::ApplicationError);

    /** Get ATU-R Interleaved Channel configuration data associated with a
        Configuration Profile */
    ChannelConf interleavedChannelConfAturGet()
        raises (itut_x780::ApplicationError);

            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectCreation)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectDeletion)


}; // end of ADSLConfigurationProfile interface




/**
```

A.1.5.4 PhyTTP

```
*/
      /*    For ADSL the PhyTTP may optionally be used to represent physical
      port inventory.  If implemented, this object should minimally provide the
      portId and a pointer to the supported ADSLLine (clientLinkEndPointerList)
      and the supporting CircuitPack (supportedByObjectList).
      It is expected that this object will be deprecated once ITU-T M.3120
      defines a CORBA object representing the port.
      */



            interface PhyTTP: itut_m3120::NetworkTP
            {

      /**   PhyTTP_ inherits the following methods from
            itut_m3120:: TP:
            operationalStateGet, alarmStatusGet, currentProblemListGet,
            alarmSeverityAssignmentProfilePointerGet,
            alarmSeverityAssignmentProfilePointerSet,
            supportedByObjectListGet

            PhyTTP inherits the following methods from
            itut_m3120:: NetworkTP:
            pointDirectionalityGet, signalIdGet


      **/

                  PortIDType phyTTPPortIDGet ()
                        raises (itut_x780::ApplicationError,
                              itut_x780::CreateError);
```

```
			MONameSetType clientLinkEndPointerListGet ()
				raises (itut_x780::ApplicationError);
			void clientLinkEndPointerListSet
				(in MONameSetType clientLinkEndPointerList)
				raises (itut_x780::ApplicationError);


			MANDATORY_NOTIFICATION(
				itut_x780::Notifications, objectCreation)
			MANDATORY_NOTIFICATION(
				itut_x780::Notifications, objectDeletion)
			MANDATORY_NOTIFICATION(
				itut_x780::Notifications, attributeValueChange)
			MANDATORY_NOTIFICATION(
				itut_x780::Notifications, stateChange)
			MANDATORY_NOTIFICATION(
				itut_x780::Notifications, communicationAlarm)


		}; // interface PhyTTP
```

/*

A.2ADSL Performance Management Module (dslf_adsl_pm_v2.idl)

```
*/
#ifndef _dslf_adsl_pm_v2_idl_
#define _dslf_adsl_pm_v2_idl_

#pragma prefix "dslforum.org"

/**
```

A.2.1 Types imported from itut_q822.1

```
*/



/***************************************************************************/
/* This module defines the interfaces for supporting Performance Management */
/* for ADSL transmission lines.                                             */
/* It is based on the CORBA Network Management Framework defined in the     */
/* ITU-T Q.822.1                               */
/***************************************************************************/

module dslf_adsl_pm_v2 {

const string moduleName = "dslf_adsl_pm_v2";

/**
IMPORTS

Interfaces applicable from Q.822.corba are:
      CurrentData_F,
      ThresholdData_F,
      HDFile_F,
      AbstractHDScanner_F,
      ListSelectionHDScanner_F,
      ScopedFiltedHDScanner_F,
      HistoryDataIterator_F.
*/
        typedef itut_x780::StringSetType StringSetType;
        typedef itut_x780::GeneralizedTimeType GeneralizedTimeType;
        typedef itut_x780ct::TimePeriodType TimePeriodType;

/**
```

A.2.2 Forward Declarations

```
*/

interface ADSLLineCurrentDataAtuc_F;
interface ADSLLineCurrentDataAtur_F;
interface ADSLChannelCurrentDataAtuc_F;
interface ADSLChannelCurrentDataAtur_F;


typedef MONameType ADSLLineCurrentDataAtucNameType;
typedef MONameType ADSLLineCurrentDataAturNameType;
typedef MONameType ADSLChannelCurrentDataAtucNameType;
typedef MONameType ADSLChannelCurrentDataAturNameType;


/**
```

A.2.3 Valuetypes, Interfaces and Methods – Façade

```
*/






valuetype ADSLLineCurrentDataValueType: itut_q822d1::CurrentDataValueType
{
      public unsigned long lossOfFrameEvents;
            // GET
      public unsigned long lossOfSignalEvents;
            // GET
      public unsigned long lossOfLinkEvents;
            // GET
      public unsigned long lossOfPowerEvents;
            // GET
      public unsigned long erroredSeconds;
            // GET
      public unsigned long severelyErroredSeconds;
            // GET
      public unsigned long unavailableSeconds;
            // GET
      public unsigned long initializationEvents;
            // GET
      public unsigned long fastRetrainAttempts;
            // GET
      public unsigned long failedFastRetrainAttempts;
            // GET
      public unsigned long codeViolations;
            // GET
      public unsigned long forwardErrorCorrections;
            // GET
      public unsigned long forwardErrorCorrectionSeconds;
            // GET
      public unsigned long lossOfSignalSeconds;
            // GET
}; // valuetype ADSLLineCurrentDataValueType


valuetype ADSLLineHistoryDataValueType: itut_q822d1::HistoryDataValueType
{
      public unsigned long lossOfFrameEvents;
            // GET
      public unsigned long lossOfSignalEvents;
            // GET
      public unsigned long lossOfLinkEvents;
            // GET
      public unsigned long lossOfPowerEvents;
            // GET
      public unsigned long erroredSeconds;
            // GET
      public unsigned long severelyErroredSeconds;
```

```
            // GET
      public unsigned long unavailableSeconds;
            // GET
      public unsigned long initializationEvents;
            // GET
      public unsigned long fastRetrainAttempts;
            // GET
      public unsigned long failedFastRetrainAttempts;
            // GET
      public unsigned long codeViolations;
            // GET
      public unsigned long forwardErrorCorrections;
            // GET
      public unsigned long forwardErrorCorrectionSeconds;
            // GET
      public unsigned long lossOfSignalSeconds;
            // GET
}; // valuetype ADSLLineHistoryDataValueType

/**
```

A.2.3.1 ADSL Line Current Data_F

```
*/

/**
This class should not be instantiated.  Individual ATU-C and ATU-R counters are
represented in their respective subclasses.
ADSL Line Current Data
Retrieves attributes or current and history data counter values for counters
within the ADSL Line ATU-C or ATU-R Protocol Monitoring grouping, namely,
lossOfFrameEvents, lossOfSignalEvents, lossOfLinkEvents, lossOfPowerEvents,
erroredSeconds, severelyErroredSeconds, unavailableSeconds,
initializationEvents, fastRetrainAttempts, failedFastRetrainAttempts,
CodeViolations, ForwardErrorCorrections, ForwardErrorCorrectionSeconds, and
LossOfSignalSeconds.


Additional inherited methods from Scanner provide for the setting of
AdministrativeState, retrieval of OperationalState, and setting of Granularity
Period.  Inherited methods from CurrentData provide for retrieval of the
SuspectFlag, retrieval of the ElapsedTime, activating or deactivating
HistoryRetention, associating threshold data with a current data instance, and
activating or deactivating the suppression of counters having all-zero counts.

Applicable methods from Scanner include: administrativeStateGet,
administrativeStateSet, operationalStateGet, granularityPeriodGet, and
granularityPeriodSet.

Applicable methods from CurrentData include: suspectIntervalFlagGet,
suspectIntervalFlagDefaultSet, elapsedTimeGet, historyRetentionGet,
historyRetentionSet, thresholdDataInstanceListGet,
thresholdDataInstanceListSet, getMostRecent, getBetween,
numSuppressedIntervalsGet, maxSuppressedIntervalsGet,
maxSuppressedIntervalsSet.

This object should support the following notifications: objectCreation,
objectDeletion, attributeValueChange, and stateChange.
```

```
A Threshold Crossing Alert (qualityOfServiceAlarm) is used to notify the
management system when any of the performance parameters exceeds a pre-set
threshold described in the associated ThresholdData object.
*/


interface ADSLLineCurrentData_F: itut_q822d1::CurrentData_F
// ADSL Line Current Data
{
/**
Count of loss of frame events; thresholded count.
*/
      unsigned long lossOfFrameEventsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of loss of signal events; thresholded count.
*/
      unsigned long lossOfSignalEventsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of loss of link events; thresholded count.
*/
      unsigned long lossOfLinkEventsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of loss of power events; thresholded count.
*/
      unsigned long lossOfPowerEventsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of errored seconds; thresholded count.
*/
      unsigned long erroredSecondsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of severely errored seconds; thresholded count.
*/
      unsigned long severelyErroredSecondsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of unavailable seconds; thresholded count.
*/
      unsigned long unavailableSecondsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of initialization events; thresholded count.
*/
      unsigned long initializationEventsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of fast retrain attempts; thresholded count.
*/
```

```
      unsigned long fastRetrainAttemptsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of failed fast retrain attempts; thresholded count.
*/
      unsigned long failedFastRetrainAttemptsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of code violations; thresholded count.
*/
      unsigned long codeViolationsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of forward error corrections; thresholded count.
*/
      unsigned long forwardErrorCorrectionsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of forward error correction seconds; thresholded count.
*/
      unsigned long forwardErrorCorrectionSecondsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of loss of signal seconds; thresholded count.
*/
      unsigned long lossOfSignalSecondsGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);

            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectCreation)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectDeletion)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, stateChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, attributeValueChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, qualityOfServiceAlarm)

}; // interface ADSLLineCurrentData_F



/**
```

A.2.3.2 ADSL Line ATU-C Current Data_F

```
*/

/**
ADSL Line ATU-C Current Data is an instantiable subclass of ADSL Line Current
Data
Retrieves attributes or current and history data counter values for counters
within the ADSL Line ATU-C Protocol Monitoring grouping, namely,
lossOfFrameEvents, lossOfSignalEvents, lossOfLinkEvents, lossOfPowerEvents,
erroredSeconds, severelyErroredSeconds, unavailableSeconds,
initializationEvents, fastRetrainAttempts, failedFastRetrainAttempts,
CodeViolations, ForwardErrorCorrections, ForwardErrorCorrectionSeconds, and
LossOfSignalSeconds.


Additional inherited methods from Scanner provide for the setting of
AdministrativeState, retrieval of OperationalState, and setting of Granularity
Period.  Inherited methods from CurrentData provide for retrieval of the
SuspectFlag, retrieval of the ElapsedTime, activating or deactivating
HistoryRetention, associating threshold data with a current data instance, and
activating or deactivating the suppression of counters having all-zero counts.

Applicable methods from Scanner include: administrativeStateGet,
administrativeStateSet, operationalStateGet, granularityPeriodGet, and
granularityPeriodSet.

Applicable methods from CurrentData include: suspectIntervalFlagGet,
suspectIntervalFlagDefaultSet, elapsedTimeGet, historyRetentionGet,
historyRetentionSet, thresholdDataInstanceListGet,
thresholdDataInstanceListSet, getMostRecent, getBetween,
numSuppressedIntervalsGet, maxSuppressedIntervalsGet,
maxSuppressedIntervalsSet.

This object should support the following notifications: objectCreation,
objectDeletion, attributeValueChange, and stateChange.

A Threshold Crossing Alert (qualityOfServiceAlarm) is used to notify the
management system when any of the performance parameters exceeds a pre-set
threshold described in the associated ThresholdData object.

A containment relationship exists between an ADSLLine and its associated
ADSLLineAtucCurrentData object.
*/

interface ADSLLineAtucCurrentData_F: ADSLLineCurrentData_F
// ADSL Line ATU-C Current Data
{
}; // interface ADSLLineAtucCurrentData_F


interface ADSLLineAtucCurrentDataFactory: itut_x780::ManagedObjectFactory
{
      itut_x780::ManagedObject create
                  (in NameBindingType nameBinding,
                        // module name containing Name Binding info.
                  in MONameType superior,
                        // Name of containing object.
                  in string reqID,
                        // Requested ID value for name, will be
                        // empty if auto-naming is to be used.
                  out MONameType name,
                        // Entire name of newly created object.
                  in StringSetType packageNameList,
                        // List of packages requested.

                  in short historyRetention,
                  in AdministrativeStateType administrativeState,
                  in TimePeriodType granularityPeriod,
                  in MONameSetType thresholdDataInstanceList,
                  in short maxSuppressedIntervals,
                  in GeneralizedTimeType periodSynchronizationTime)
                  raises (itut_x780::ApplicationError,
                        itut_x780::CreateError);

}; // interface ADSLLineAtucCurrentDataFactory




/**
```

A.2.3.3 ADSL Line ATU-R Current Data_F

```
*/
/**
ADSL Line ATU-R Current Data is an instantiable subclass of ADSL Line Current
Data
Retrieves attributes or current and history data counter values for counters
within the ADSL Line ATU-R Protocol Monitoring grouping, namely,
lossOfFrameEvents, lossOfSignalEvents, lossOfLinkEvents, lossOfPowerEvents,
erroredSeconds, severelyErroredSeconds, unavailableSeconds,
initializationEvents, fastRetrainAttempts, failedFastRetrainAttempts,
CodeViolations, ForwardErrorCorrections, ForwardErrorCorrectionSeconds, and
LossOfSignalSeconds.


Additional inherited methods from Scanner provide for the setting of
AdministrativeState, retrieval of OperationalState, and setting of Granularity
Period.  Inherited methods from CurrentData provide for retrieval of the
SuspectFlag, retrieval of the ElapsedTime, activating or deactivating
HistoryRetention, associating threshold data with a current data instance, and
activating or deactivating the suppression of counters having all-zero counts.

Applicable methods from Scanner include: administrativeStateGet,
administrativeStateSet, operationalStateGet, granularityPeriodGet, and
granularityPeriodSet.

Applicable methods from CurrentData include: suspectIntervalFlagGet,
suspectIntervalFlagDefaultSet, elapsedTimeGet, historyRetentionGet,
historyRetentionSet, thresholdDataInstanceListGet,
thresholdDataInstanceListSet, getMostRecent, getBetween,
numSuppressedIntervalsGet, maxSuppressedIntervalsGet,
maxSuppressedIntervalsSet.

This object should support the following notifications: objectCreation,
objectDeletion, attributeValueChange, and stateChange.

A Threshold Crossing Alert (qualityOfServiceAlarm) is used to notify the
management system when any of the performance parameters exceeds a pre-set
threshold described in the associated ThresholdData object.

A containment relationship exists between an ADSLLine and its associated
ADSLLineAturCurrentData object.
*/

interface ADSLLineAturCurrentData_F: ADSLLineCurrentData_F
// ADSL Line ATU-R Current Data
{
}; // interface ADSLLineAturCurrentData_F


interface ADSLLineAturCurrentDataFactory: itut_x780::ManagedObjectFactory
{
      itut_x780::ManagedObject create
                  (in NameBindingType nameBinding,
                        // module name containing Name Binding info.
                  in MONameType superior,
                        // Name of containing object.
                  in string reqID,
                        // Requested ID value for name, will be
                        // empty if auto-naming is to be used.
                  out MONameType name,
                        // Entire name of newly created object.
                  in StringSetType packageNameList,
                        // List of packages requested.

                  in short historyRetention,
                  in AdministrativeStateType administrativeState,
                  in TimePeriodType granularityPeriod,
                  in MONameSetType thresholdDataInstanceList,
                  in short maxSuppressedIntervals,
                  in GeneralizedTimeType periodSynchronizationTime)
                  raises (itut_x780::ApplicationError,
                        itut_x780::CreateError);

}; // interface ADSLLineAturCurrentDataFactory

valuetype ADSLChannelCurrentDataValueType: itut_q822d1::CurrentDataValueType
{
      public unsigned long transmittedBlocks;
            // GET
      public unsigned long receivedBlocks;
            // GET
      public unsigned long correctedBlocks;
            // GET
      public unsigned long uncorrectedBlocks;
            // GET
}; // valuetype ADSLChannelCurrentDataValueType

valuetype ADSLChannelHistoryDataValueType: itut_q822d1::HistoryDataValueType
{
      public unsigned long transmittedBlocks;
            // GET
      public unsigned long receivedBlocks;
            // GET
      public unsigned long correctedBlocks;
            // GET
      public unsigned long uncorrectedBlocks;
            // GET
}; // valuetype ADSLChannelHistoryDataValueType

/**
```

A.2.3.4 ADSL Channel Current Data_F

```
*/

/**
```

```
This class should not be instantiated.  Individual ATU-C and ATU-R counters are
represented in their respective subclasses.

ADSL Channel Current Data
Retrieves attributes or current and history data counter values for counters
within the ADSL Channel ATU-C and ATU-R Protocol Monitoring grouping, namely,
transmittedBlocks, receivedBlocks, correctedBlocks, and uncorrectedBlocks.

Additional inherited methods from Scanner provide for the setting of
AdministrativeState, retrieval of OperationalState, and setting of Granularity
Period.  Inherited methods from CurrentData provide for retrieval of the
SuspectFlag, retrieval of the ElapsedTime, activating or deactivating
HistoryRetention, associating threshold data with a current data instance, and
activating or deactivating the suppression of counters having all-zero counts.

Applicable methods from Scanner include: administrativeStateGet,
administrativeStateSet, operationalStateGet, granularityPeriodGet, and
granularityPeriodSet.

Applicable methods from CurrentData include: suspectIntervalFlagGet,
suspectIntervalFlagDefaultSet, elapsedTimeGet, historyRetentionGet,
historyRetentionSet, thresholdDataInstanceListGet,
thresholdDataInstanceListSet, getMostRecent, getBetween,
numSuppressedIntervalsGet, maxSuppressedIntervalsGet,
maxSuppressedIntervalsSet.

This object should support the following notifications: objectCreation,
objectDeletion, attributeValueChange, and stateChange.

A Threshold Crossing Alert (qualityOfServiceAlarm) is used to notify the
management system when any of the performance parameters exceeds a pre-set
threshold described in the associated ThresholdData object.

*/

interface ADSLChannelCurrentData_F: itut_q822d1::CurrentData_F
// ADSL Channel Current Data
{
/**
Count of transmitted blocks.
*/
      unsigned long transmittedBlocksGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of received blocks.
*/
      unsigned long receivedBlocksGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of corrected blocks.
*/
      unsigned long correctedBlocksGet
            (in MONameType name)
            raises (itut_x780::ApplicationError);
/**
Count of uncorrected blocks; thresholded count.
*/
      unsigned long uncorrectedBlocksGet
```

```
            (in MONameType name)
            raises (itut_x780::ApplicationError);

            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectCreation)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectDeletion)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, stateChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, attributeValueChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, qualityOfServiceAlarm)

}; // interface ADSLChannelCurrentData_F


/**
```

A.2.3.5 ADSL Channel ATU-C Current Data_F

```
*/

/**
ADSL Channel ATU-C Current Data is an instantiable subclass of ADSL Channel
Current Data
Retrieves attributes or current and history data counter values for counters
within the ADSL Channel ATU-C Protocol Monitoring grouping, namely,
transmittedBlocks, receivedBlocks, correctedBlocks, and uncorrectedBlocks.

Additional inherited methods from Scanner provide for the setting of
AdministrativeState, retrieval of OperationalState, and setting of Granularity
Period.  Inherited methods from CurrentData provide for retrieval of the
SuspectFlag, retrieval of the ElapsedTime, activating or deactivating
HistoryRetention, associating threshold data with a current data instance, and
activating or deactivating the suppression of counters having all-zero counts.

Applicable methods from Scanner include: administrativeStateGet,
administrativeStateSet, operationalStateGet, granularityPeriodGet, and
granularityPeriodSet.

Applicable methods from CurrentData include: suspectIntervalFlagGet,
suspectIntervalFlagDefaultSet, elapsedTimeGet, historyRetentionGet,
historyRetentionSet, thresholdDataInstanceListGet,
thresholdDataInstanceListSet, getMostRecent, getBetween,
numSuppressedIntervalsGet, maxSuppressedIntervalsGet,
maxSuppressedIntervalsSet.

This object should support the following notifications: objectCreation,
objectDeletion, attributeValueChange, and stateChange.

A Threshold Crossing Alert (qualityOfServiceAlarm) is used to notify the
management system when any of the performance parameters exceeds a pre-set
threshold described in the associated ThresholdData object.

A containment relationship exists between an ADSLChannel and its associated
ADSLChannelAtucCurrentData object.
*/

interface ADSLChannelAtucCurrentData_F: ADSLChannelCurrentData_F
```

```
// ADSL Channel ATU-C Current Data
{
}; // interface ADSLChannelAtucCurrentData_F


interface ADSLChannelAtucCurrentDataFactory: itut_x780::ManagedObjectFactory
{
      itut_x780::ManagedObject create
                  (in NameBindingType nameBinding,
                        // module name containing Name Binding info.
                  in MONameType superior,
                        // Name of containing object.
                  in string reqID,
                        // Requested ID value for name, will be
                        // empty if auto-naming is to be used.
                  out MONameType name,
                        // Entire name of newly created object.
                  in StringSetType packageNameList,
                        // List of packages requested.

                  in short historyRetention,
                  in AdministrativeStateType administrativeState,
                  in TimePeriodType granularityPeriod,
                  in MONameSetType thresholdDataInstanceList,
                  in short maxSuppressedIntervals,
                  in GeneralizedTimeType periodSynchronizationTime)
                  raises (itut_x780::ApplicationError,
                        itut_x780::CreateError);

}; // interface ADSLChannelAtucCurrentDataFactory

/**
```

A.2.3.6 ADSL Channel ATU-R Current Data_F

```
*/


/**
ADSL Channel ATU-R Current Data is an instantiable subclass of ADSL Line
Current Data.
Retrieves attributes or current and history data counter values for counters
within the ADSL Channel ATU-R Protocol Monitoring grouping, namely,
transmittedBlocks, receivedBlocks, correctedBlocks, and uncorrectedBlocks.

Additional inherited methods from Scanner provide for the setting of
AdministrativeState, retrieval of OperationalState, and setting of Granularity
Period.  Inherited methods from CurrentData provide for retrieval of the
SuspectFlag, retrieval of the ElapsedTime, activating or deactivating
HistoryRetention, associating threshold data with a current data instance, and
activating or deactivating the suppression of counters having all-zero counts.

Applicable methods from Scanner include: administrativeStateGet,
administrativeStateSet, operationalStateGet, granularityPeriodGet, and
granularityPeriodSet.

Applicable methods from CurrentData include: suspectIntervalFlagGet,
suspectIntervalFlagDefaultSet, elapsedTimeGet, historyRetentionGet,
historyRetentionSet, thresholdDataInstanceListGet,
thresholdDataInstanceListSet, getMostRecent, getBetween,
numSuppressedIntervalsGet, maxSuppressedIntervalsGet,
maxSuppressedIntervalsSet.

This object should support the following notifications: objectCreation,
objectDeletion, attributeValueChange, and stateChange.

A Threshold Crossing Alert (qualityOfServiceAlarm) is used to notify the
management system when any of the performance parameters exceeds a pre-set
threshold described in the associated ThresholdData object.

A containment relationship exists between an ADSLChannel and its associated
ADSLChannelAturCurrentData object.
*/

interface ADSLChannelAturCurrentData_F: ADSLChannelCurrentData_F
// ADSL Channel ATU-R Current Data
{
}; // interface ADSLChannelAturCurrentData_F


interface ADSLChannelAturCurrentDataFactory: itut_x780::ManagedObjectFactory
{
      itut_x780::ManagedObject create
                  (in NameBindingType nameBinding,
                        // module name containing Name Binding info.
                  in MONameType superior,
                        // Name of containing object.
                  in string reqID,
                        // Requested ID value for name, will be
                        // empty if auto-naming is to be used.
                  out MONameType name,
                        // Entire name of newly created object.
                  in StringSetType packageNameList,
                        // List of packages requested.

                  in short historyRetention,
                  in AdministrativeStateType administrativeState,
                  in TimePeriodType granularityPeriod,
                  in MONameSetType thresholdDataInstanceList,
                  in short maxSuppressedIntervals,
                  in GeneralizedTimeType periodSynchronizationTime)
                  raises (itut_x780::ApplicationError,
                        itut_x780::CreateError);

}; // interface ADSLChannelAturCurrentDataFactory
```

```
/**
```

A.2.4 Interfaces and Methods – Fine Grained

The behaviour of the Fine-Grained interfaces are identical to the corresponding Façade grained interfaces. Therefore, comments are not included in the fine-grained interfaces. Readers are referred to the Façade interface in Section A.2.3 for the behaviour of the fine-grained interface.

This section can be omitted from IDL if a management system only supports Façade grained interface.

```
*/
```

```
/**
```

A.2.4.1 ADSL Line Current Data

```
*/



interface ADSLLineCurrentData: itut_q822d1::CurrentData
// ADSL Line Current Data
{
/**
Count of loss of frame events; thresholded count.
*/
      unsigned long lossOfFrameEventsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of loss of signal events; thresholded count.
*/
      unsigned long lossOfSignalEventsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of loss of link events; thresholded count.
*/
      unsigned long lossOfLinkEventsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of loss of power events; thresholded count.
*/
      unsigned long lossOfPowerEventsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of errored seconds; thresholded count.
*/
      unsigned long erroredSecondsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of severely errored seconds; thresholded count.
*/
      unsigned long severelyErroredSecondsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of unavailable seconds; thresholded count.
*/
      unsigned long unavailableSecondsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of initialization events; thresholded count.
*/
      unsigned long initializationEventsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of fast retrain attempts; thresholded count.
*/
      unsigned long fastRetrainAttemptsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of failed fast retrain attempts; thresholded count.
*/
      unsigned long failedFastRetrainAttemptsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of code violations; thresholded count.
*/
      unsigned long codeViolationsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of forward error corrections; thresholded count.
*/
      unsigned long forwardErrorCorrectionsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of forward error correction seconds; thresholded count.
*/
      unsigned long forwardErrorCorrectionSecondsGet ()
            raises (itut_x780::ApplicationError);
/**
Count of loss of signal seconds; thresholded count.
*/
      unsigned long lossOfSignalSecondsGet ()
            raises (itut_x780::ApplicationError);

            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectCreation)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectDeletion)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, stateChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, attributeValueChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, qualityOfServiceAlarm)

}; // interface ADSLLineCurrentData


/**
```

A.2.4.2 ADSL Line ATU-C Current Data

```
*/


interface ADSLLineAtucCurrentData: ADSLLineCurrentData
// ADSL Line ATU-C Current Data
{
}; // interface ADSLLineAtucCurrentData




/**
```

A.2.4.3 ADSL Line ATU-R Current Data

```
*/

interface ADSLLineAturCurrentData: ADSLLineCurrentData
// ADSL Line ATU-R Current Data
{
}; // interface ADSLLineAturCurrentData
```

```
/**
```

A.2.4.4 ADSL Channel Current Data

```
*/


interface ADSLChannelCurrentData: itut_q822d1::CurrentData
// ADSL Channel Current Data
{
/**
Count of transmitted blocks.
*/
      unsigned long transmittedBlocksGet ()
            raises (itut_x780::ApplicationError);
/**
Count of received blocks.
*/
      unsigned long receivedBlocksGet ()
            raises (itut_x780::ApplicationError);
/**
Count of corrected blocks.
*/
      unsigned long correctedBlocksGet ()
            raises (itut_x780::ApplicationError);
/**
Count of uncorrected blocks; thresholded count.
*/
      unsigned long uncorrectedBlocksGet ()
            raises (itut_x780::ApplicationError);

            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectCreation)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, objectDeletion)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, stateChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, attributeValueChange)
            MANDATORY_NOTIFICATION(
                  itut_x780::Notifications, qualityOfServiceAlarm)

}; // interface ADSLChannelCurrentData


/**
```

A.2.4.5 ADSL Channel ATU-C Current Data

```
*/


interface ADSLChannelAtucCurrentData: ADSLChannelCurrentData
// ADSL Channel ATU-C Current Data
{
}; // interface ADSLChannelAtucCurrentData
```

```
/**
```

A.2.4.6 ADSL Channel ATU-R Current Data

```
*/


interface ADSLChannelAturCurrentData: ADSLChannelCurrentData
// ADSL Channel ATU-R Current Data
{
}; // interface ADSLChannelAturCurrentData



}; // end of dslf_adsl_pm_v2 module

/*****************************************************************************/
#endif // end of ifndef _dslf_adsl_pm_v2_
```

```
/**
```

A.3ADSL Namebindings Module (dslf_adsl_namebindings.idl)

```
 */

module dslf_adsl_namebindings
{

/**
```

A.3.1 ADSLLine_ManagedElement

```
*/

            module ADSLLine_ManagedElement
            {
                  const string      superiorClass =
                        "itut_m3120::ManagedElement";
                  const boolean     superiorSubclassesAllowed = TRUE;
                  const string      subordinateClass =
                        "dslf_adsl_v2::ADSLLine";
                  const boolean     subordinateSubclassesAllowed = TRUE;
                  const boolean     managerCreatesAllowed = TRUE;
                  const DeletePolicyType deletePolicy =
                        itut_x780::deleteOnlyIfNoContainedObjects;
                  const string      kind = "ADSLLine";
            }; // module ADSLLine_ManagedElement

/**
```

A.3.2 ADSLChannel_ADSLLine

```
*/

            module ADSLChannel_ADSLLine
            {
                  const string      superiorClass =
                        "dslf_adsl_v2::ADSLLine";
                  const boolean     superiorSubclassesAllowed = TRUE;
                  const string      subordinateClass =
                        "dslf_adsl_v2::ADSLChannel";
                  const boolean     subordinateSubclassesAllowed = TRUE;
                  const boolean     managerCreatesAllowed = TRUE;
                  const DeletePolicyType deletePolicy =
                        itut_x780::deleteOnlyIfNoContainedObjects;
                  const string      kind = "ADSLChannel";
            }; // module ADSLChannel_ADSLLine

/**
```

A.3.3 ADSLConfigurationProfile_ManagedElement

```
*/

            module ADSLConfigurationProfile_ManagedElement
            {
                  const string      superiorClass =
                        "itut_m3120::ManagedElement";
                  const boolean     superiorSubclassesAllowed = TRUE;
                  const string      subordinateClass =
                        "dslf_adsl_v2::ADSLConfigurationProfile";
                  const boolean     subordinateSubclassesAllowed = TRUE;
                  const boolean     managerCreatesAllowed = TRUE;
                  const DeletePolicyType deletePolicy =
                        itut_x780::deleteOnlyIfNoContainedObjects;
                  const string      kind = "ADSLConfigurationProfile";
            }; // module ADSLConfigurationProfile_ManagedElement

/**
```

A.3.4 PhyTTP_ManagedElement

```
*/
              module PhyTTP_ManagedElement
              {
                     const string   superiorClass =
                            "itut_m3120::ManagedElement";
                     const boolean  superiorSubclassesAllowed = TRUE;
                     const string   subordinateClass =
                            "dslf_adsl_v2::PhyTTP";
                     const boolean  subordinateSubclassesAllowed = TRUE;
                     const boolean  managerCreatesAllowed = FALSE;
                     const DeletePolicyType deletePolicy =
                            itut_x780::notDeletable;
                     const string   kind = "PhyTTP";
              }; // module PhyTTP_ManagedElement

/**
```

A.3.5 ADSLLineAtucCurrentData_ADSLLine

```
*/

            module ADSLLineAtucCurrentData_ADSLLine
            {
                  const string      superiorClass =
                        "dslf_adsl_v2::ADSLLine";
                  const boolean     superiorSubclassesAllowed = TRUE;
                  const string      subordinateClass =
                        "dslf_adsl_pm_v2::ADSLLineAtucCurrentData";
                  const boolean     subordinateSubclassesAllowed = TRUE;
                  const boolean     managerCreatesAllowed = TRUE;
                  const DeletePolicyType deletePolicy =
                        itut_x780::deleteOnlyIfNoContainedObjects;
                  const string      kind = "ADSLLineAtucCurrentData";
            }; // module ADSLLineAtucCurrentData_ADSLLine

/**
```

A.3.6 ADSLLineAturCurrentData_ADSLLine

```
*/

            module ADSLLineAturCurrentData_ADSLLine
            {
                  const string      superiorClass =
                        "dslf_adsl_v2::ADSLLine";
                  const boolean     superiorSubclassesAllowed = TRUE;
                  const string      subordinateClass =
                        "dslf_adsl_pm_v2::ADSLLineAturCurrentData";
                  const boolean     subordinateSubclassesAllowed = TRUE;
                  const boolean     managerCreatesAllowed = TRUE;
                  const DeletePolicyType deletePolicy =
                        itut_x780::deleteOnlyIfNoContainedObjects;
                  const string      kind = "ADSLLineAturCurrentData";
            }; // module ADSLLineAturCurrentData_ADSLLine

/**
```

A.3.7 ADSLChannelAtucCurrentData_ADSLChannel

```
*/

            module ADSLChannelAtucCurrentData_ADSLChannel
            {
                  const string      superiorClass =
                        "dslf_adsl_v2::ADSLChannel";
                  const boolean     superiorSubclassesAllowed = TRUE;
                  const string      subordinateClass =
                        "dslf_adsl_pm_v2::ADSLChannelAtucCurrentData";
                  const boolean     subordinateSubclassesAllowed = TRUE;
                  const boolean     managerCreatesAllowed = TRUE;
                  const DeletePolicyType deletePolicy =
                        itut_x780::deleteOnlyIfNoContainedObjects;
                  const string      kind = "ADSLChannelAtucCurrentData";
            }; // module ADSLChannelAtucCurrentData_ADSLChannel

/**
```

A.3.8 ADSLChannelAturCurrentData_ADSLChannel

```
*/

            module ADSLChannelAturCurrentData_ADSLChannel
            {
                  const string      superiorClass =
                        "dslf_adsl_v2::ADSLChannel";
                  const boolean     superiorSubclassesAllowed = TRUE;
                  const string      subordinateClass =
                        "dslf_adsl_pm_v2::ADSLChannelAturCurrentData";
                  const boolean     subordinateSubclassesAllowed = TRUE;
                  const boolean     managerCreatesAllowed = TRUE;
                  const DeletePolicyType deletePolicy =
                        itut_x780::deleteOnlyIfNoContainedObjects;
                  const string      kind = "ADSLChannelAturCurrentData";
            }; // module ADSLChannelAturCurrentData_ADSLChannel


}; // module NameBinding



}; // end of dslf_adsl_v2 module
/*****************************************************************************/

#endif // end of ifndef _dslf_adsl_v2_idl_
```

Appendix B: Object Naming Guidelines

This appendix provides suggested object naming guidelines for the DSL CORBA IDL MIB. Such guidelines will promote EMS and NMS interoperability. The naming guidelines, in the table below, provide a name syntax for each CORBA object. The name syntax should be used as the id of the name component of the object.

Syntax:
"text" – text inside quotation should appear as is
| - means OR
<*type*> - identifies a type or category
[optional item] – indicates an optional item
{repetitive item} – indicates an item that may appear zero or more times

Defined Types used:
<*String*> ::= {any_character}
<*null*> - indicates a null string

Note: asterisk "*" is used as a field delimiter.

Table B-1. Object Naming Guidelines

Object	**Name Syntax** (NameComponent.id for object)
dslf_adsl_v2:ADSLLine	"ManagedElementName="<*String*>"*" "Bay="<*String*>"*" "Shelf="<*String*>"*" "Slot="<*String*>"*" "Port="<*String*>
dslf_adsl_v2:ADSLChannel	"ManagedElementName="<*String*>"*" "Bay="<*String*>"*" "Shelf="<*String*>"*" "Slot="<*String*>"*" "Port="<*String*>"*" "ADSLChannelName="<*String*>
dslf_adsl_v2:ADSLConfigurationPro file	"ManagedElementName="<*String*>"*" "ProfileName="<*String*>
dslf_adsl_v2:PhyTTP	"ManagedElementName="<*String*>"*" "Bay="<*String*>"*" "Shelf="<*String*>"*" "Slot="<*String*>"*" "Port="<*String*>
dslf_adsl_v2:	<*ADSLLineName*>"*"

Object	**Name Syntax** (NameComponent.id for object)
ADSLLineAtucCurrentData	"CurrentDataName="<*String*>
dslf_adsl_v2: ADSLLineAturCurrentData	<*ADSLLineName*>"*" "CurrentDataName="<*String*>
dslf_adsl_v2: ADSLChannelAtucCurrentData	<*ADSLChannelName*>"*" "CurrentDataName="<*String*>
dslf_adsl_v2: ADSLChannelAturCurrentData	<*ADSLChannelName*>"*" "CurrentDataName="<*String*>