ICS 33 040
M 19

YD

中华人民共和国通信行业标准

YD/T 1665-2007

数字用户线路（DSL）网络管理接口技术要求

Technical Specification for DSL Network Management Interface

2007-07-20 发布　　2007-12-01 实施

中华人民共和国信息产业部　发布

目　次

前　言

本标准结合我国的实际情况修改，采用了以下标准：

1．ADSL Forum TR-035: “Protocol Independent Object Model for ADSL EMS-NMS Interface”. Mar. 2000；

2．DSL Forum TR-030: “ADSL EMS to NMS Functional Requirements”, Feb. 2000。

其中，在信息模型方面主要参考了 ADSL Forum TR-035: “Protocol Independent Object Model for ADSL EMS-NMS Interface”. Mar. 2000。不同之处在于：根据实际的 DSL 组网情况以及本标准的范围，去除了 ATM 的管理实体。在功能需求方面主要参考了 DSL Forum TR-030: “ADSL EMS to NMS Functional Requirements”, Feb. 2000。不同之处在于：根据实际本标准的范围除去了 ATM 相关功能和 ATM 层的相关功能（类似 VP/VC 等交叉连接）。公共管理接口功能需求方面去除了对逻辑网元方面的需求，添加了通知管理功能需求。在告警管理功能需求方面添加了一些可选的告警功能需求，如确认/去确认告警和清除告警等。安全管理功能需求方面根据实际需要变成可选部分。

本标准中的附录 A 为资料性附录。

本标准由中国通信标准化协会提出并归口

本标准起草单位：中兴通讯股份有限公司、上海贝尔阿尔卡特股份有限公司、北京西门子通信网络股份有限公司

本标准主要起草人：孙　鸣、申山宏、苏春山、万光华、苏鹏、朱建华

数字用户线路（DSL）网络管理接口技术要求

1 范围

本标准描述了DSL通信网的网络结构和网络管理结构，规定了DSL通信网网络管理中NMS和EMS之间的接口，包括接口位置、功能需求、信息模型以及协议有关的接口定义。

DSL通信网网络管理接口功能需求包括公共管理接口功能需求、配置管理接口功能需求、性能管理接口功能需求和故障管理功能接口需求。

本标准定义了ADSL、VDSL、SHDSL 3种用户线技术的网络管理接口要求。

本标准适用于中国宽带网络的DSL网络管理接口。

2 规范性引用文件

下列文件中的条款通过本标准的引用而成为本标准的条款。凡是注日期的引用文件，其随后所有的修改单（不包括勘误的内容）或修订版均不适用于本标准。然而，鼓励根据本标准达成协议的各方研究是否可使用这些文件的最新版本。凡是不注日期的引用文件，其最新版本适用于本标准。

ITU-T Recommendation M.3010: “Principles for a Telecommunications Management Network” ,April，2005

ITU-T Recommendation M.3100: “Generic Network Information Model” ,2000

ADSL ForumWT-041: ”ADSL EMS to NMS Functional Requirements”,August 1999.

DSL Forum TR-030: “ADSL EMS to NMS Functional Requirements”, Feb. 2000

ADSL Forum TR-005: ”ADSL Network Element Management”.

DSL Forum TR-066: “ADSL Network Element Management (Update to TR-005)”, March 2004.

ADSL Forum adslf99_198: “Protocol Independent Model for ADSL EMSNMS Interface”, August, 1999.

ADSL Forum TR-012: “ Broadband Service Architecture for Access to Legacy Data Networks over ADSL”.

ADSL Forum TR-035: “Protocol Independent Object Model for ADSL EMS-NMS Interface”. Mar. 2000

ADSL Forum TR-041: “CORBA Specification forADSL EMS-NMS Interface”. Jun. 2001

DSL Forum TR-050: “CORBA v2 for ADSL EMS-NMS Interface”, Apr. 2002

RFC3728: “ Definitions of Managed Objects for Very High Speed Digital Subscriber Lines (VDSL)”, 2004

RFC3276: “ Definitions of Managed Objects for High Bit-Rate DSL - 2nd generation (HDSL2) and Single-Pair High-Speed Digital Subscriber Line (SHDSL) Lines”, 1999.

3 术语、定义和缩略语

3.1 术语和定义

下列术语和定义适用于本标准。

3.1.1 网元管理系统（Element Management System，EMS）

EMS：网元管理系统，通常由设备提供商提供，能够管理一个厂商的多种网络设备。

3.1.2 网元设备（Network Element，NE）

NE：指的是由 EMS 管理的各种网络设备，例如 *x*DSL 设备。

3.1.3 网络管理系统（Network Management System，NMS）

NMS：对由多种类型，多个厂商的设备组成的网络进行端到端管理的系统。NMS 并不直接管理设备，而是依赖于 EMS 进行管理。一个 NMS 系统可能与一个或多个业务管理系统相接。NMS 也可能具备一部分 EMS 的功能，可以针对单个网元进行管理；也可以只具备网络层的功能，管理一个或多个 EMS。

3.1.4 通知服务 （Notification Service）

接口侧的通知分发服务提供通知控制和转发功能。

3.2 缩略语

下列缩略语适用于本标准：

EMS	Element Management System	网元管理系统
NMS	Network Management System	网络管理系统
TMN	Telecommunications Management Network	电信管理网
ATU-C	ADSL Transmission Unit at the Central Office end.	ADSL在局端的传输单元
ATU-R	ADSL Transmission Unit at the Remote end.	ADSL在用户端的传输单元
ATM	Asynchronous Transfer Mode	异步传输模式
VTUC	VDSL Transmission Unit at the Central Office end.	VDSL在局端的传输单元
VTUR	VDSL Transmission Unit at the Remote end.	VDSL在用户端的传输单元
STUC	SHDSL Transmission Unit at the Central Office end.	SHDSL在局端的传输单元
STUR	SHDSL Transmission Unit at the Remote end.	SHDSL在用户端的传输单元
SNR	Signal to Noise Ratio	信噪比
Mgn	Margin	裕度
CRC	Cyclic Redundancy Check	循环冗余校验
Atn	Attenuation	衰减
ES	Errored Seconds	误码秒数
SES	Severely Errored Seconds	严重误码秒数
UAS	Unavailable Seconds	不可用秒数
LOSW	Loss of Sync Word	同步字丢失
LOSWS	LOSW seconds	同步字丢失秒数
Los	Loss of Signal	信号丢失
Loss	Los Seconds	信号丢失秒数
Lof	Loss of Frame	帧丢失
Lofs	Lof Seconds	帧丢失秒数
Lol	Loss of Link	链路丢失
Lols	Lol Seconds	链路丢失秒数
Lpr	Loss of Power	电源丢失

Lprs	Lpr Seconds	电源丢失秒数

4 网络管理结构

这节主要描述了一个典型 DSL 接入网宽带网络管理架构。在一个宽带运维架构中，一个 NMS 为一个多供应商，多种技术构成的网络提供端到端的管理。这个架构既平衡了网络提供商的不同的 EMS 产品，同时也支持上层运维支撑系统的网管接口。

图 1 描绘了一个 ADSL 的 EMS 和宽带 NMS 共存的一个典型环境。NMS 管理的网络可以包括 DSL 以及其他宽带类型的网元和 EMS 系统。

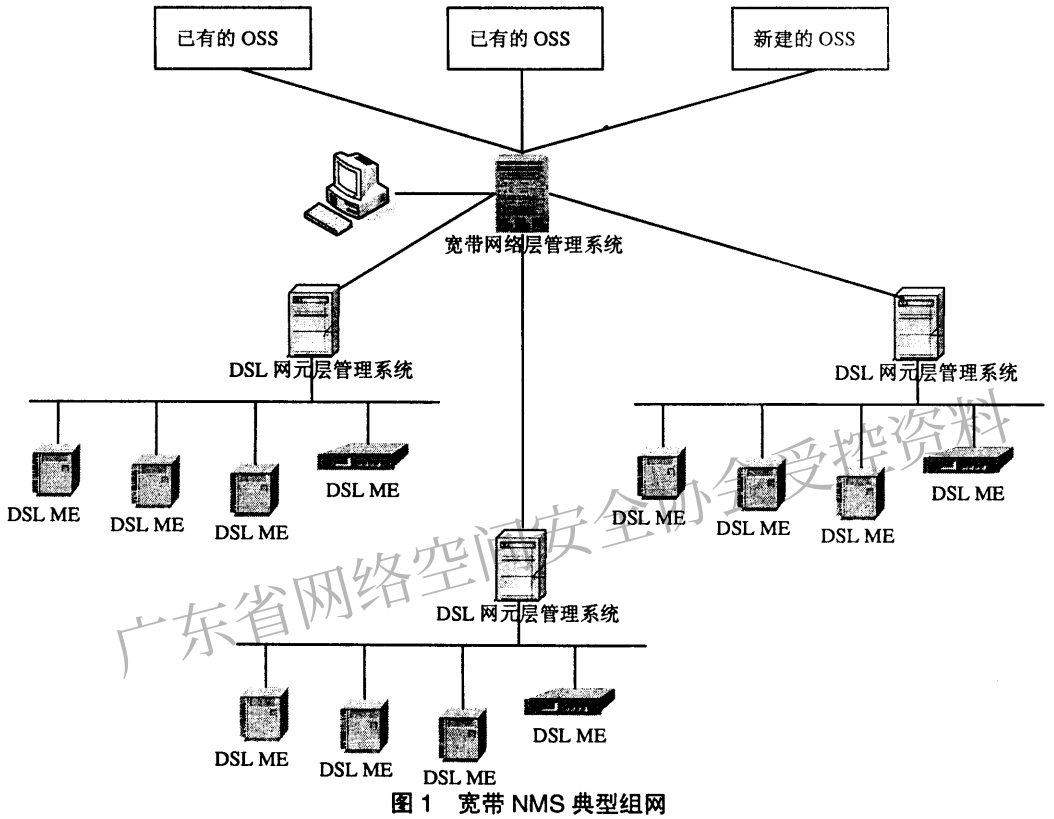


图 1 宽带 NMS 典型组网

5 DSL 通信网网络管理接口定义

5.1 接口位置

在DSL通信网网络管理中，有网络管理系统（NMS）和网元管理系统（EMS）。它们之间的网络管理接口称为北向接口，下文简称为Itf-N，即本标准中所指的接口。

Itf-N的位置如图2所示。

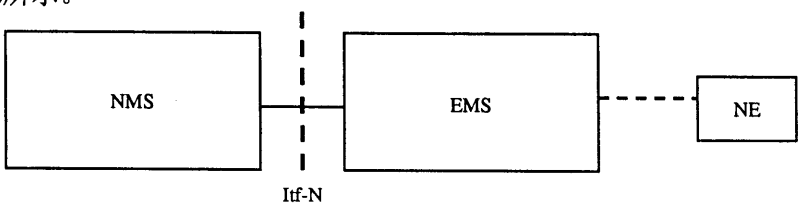


图 2 Itf-N 接口位置

在图2中，NE泛指设备商提供的DSL通信设备，可以是单个设备，也可以是多个设备；EMS是由设备厂商自行提供的网元管理系统，可以对本厂商的DSL设备进行配置、操作和维护等；NMS是网络管理系统，能够管理不同设备厂商的DSL设备。

如图2所示，Itf-N位于NMS与EMS之间，由EMS向NMS提供北向接口，在这种情况下，EMS和NE之间的接口为设备内部接口，不在本规范定义的范围内。

在网络管理中，相互交互的两个实体分别承担了两个角色——管理者（Manager）或是代理者（Agent）。管理者的任务是发送管理命令并接收代理者发来的通知；代理者的任务是直接管理有关的管理对象，接收管理者发来的管理命令并向管理者返回操作响应（确认型操作时），也可在需要时主动向管理者发出通知。图3描述了管理者和代理者间的基本关系。对应图2，可以认为NMS承担着管理者的职责，EMS承担着代理者的职责。

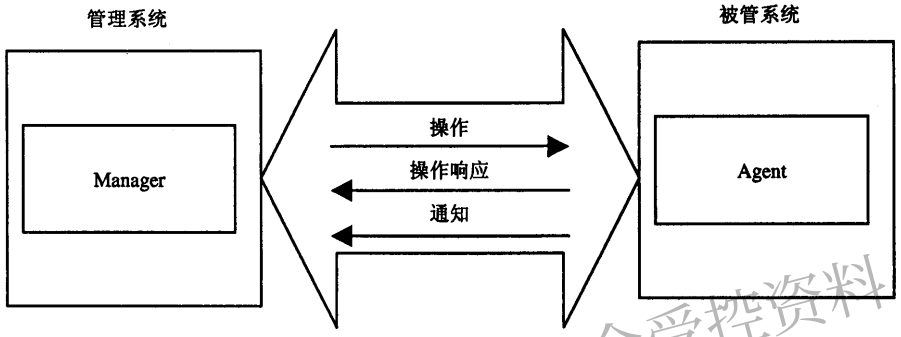


图 3　管理者与代理者关系示意

5.2　接口含义

网络管理接口指的是Manager与Agent之间进行管理信息交互的通道，包含以下3层含义：

第一，支持Manager与Agent之间进行通信的通信协议栈。这是网络管理接口应具备的基本功能，即网络管理接口应具备底层传输协议，以完成数据的传输。

第二，应用层网络管理协议。网络管理接口应对应用层网络管理协议进行定义，以完成Manager和Agent之间交换端到端的管理信息。如定义Manager从Agent 获取信息的方式，包括管理动作和应答的交互、上报通知的方式等。

第三，统一的管理信息模型。网络管理接口应对Manager和Agent之间传递的管理信息进行定义，使得双方对管理信息的语义及语法有统一的认识。通过网管接口交互的管理信息，包括管理控制信息和网络资源信息，均含语义定义和语法定义。

上述3部分网络管理接口含义的关系如图4所示。

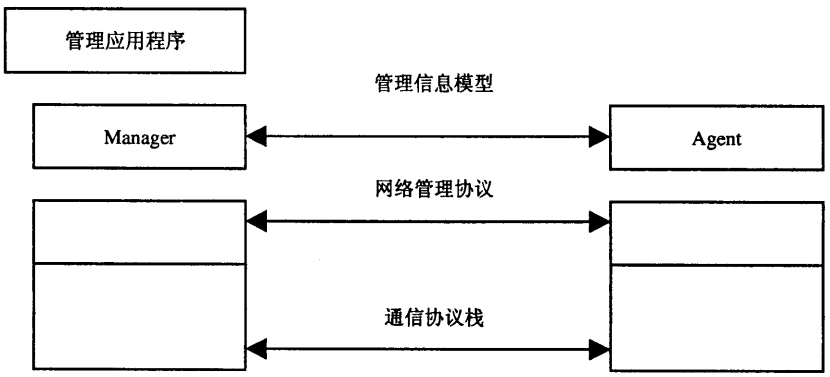


图 4　接口含义及其关系示意

5.3 接口定义内容

根据接口含义的要求，本规范在定义DSL网络管理的Itf-N时，将从如下两个方面进行定义：

（1）接口管理功能

由于接口管理信息模型的定义是基于接口管理功能需求，并为其服务的，因此在定义接口管理信息之前，需要对接口管理功能作出准确的定义。

Itf-N的接口管理功能需求可以分为不同的管理域，如配置管理接口需求、性能管理接口需求、故障管理接口需求和其他管理接口需求等，对应的Itf-N由一簇管理接口组成，如配置管理接口、性能管理接口、故障管理接口和其他管理接口等，如图5所示。

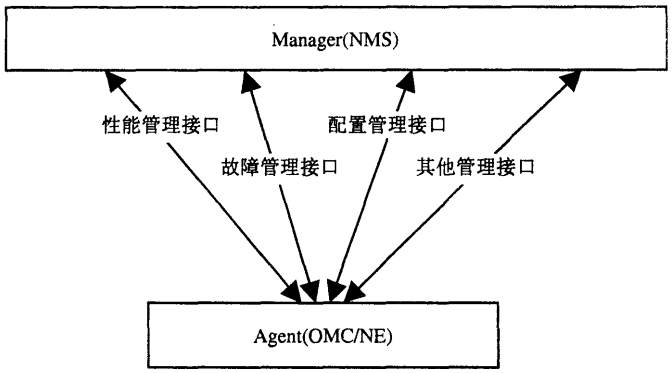


图 5 接口管理功能示例

（2）接口管理信息模型，根据接口管理功能的要求，本技术规范还将定义具体的通过网管接口交互的管理信息，包括管理控制信息和网络资源信息（见上文介绍），含语义定义和语法定义。在本规范中，管理控制信息以管理域为粒度进行定义，如针对配置管理接口需求，定义一个相应的配置管理接口控制对象；根据性能管理需求，定义一个相应的性能管理接口控制对象。这样的控制对象称为管理域控制对象。

6 接口管理功能需求

DSL 网络管理接口功能需求由公共管理接口功能需求、配置管理接口功能需求、通知管理接口功能需求、故障管理接口功能需求、性能管理接口功能需求和安全管理接口功能需求（可选）组成。

6.1 公共管理接口功能需求

本小节主要描述 DSL 网络管理接口功能需求中的公共管理接口功能需求，公共管理接口功能需求主要描述 Manager 和 Agent 之间接口的通用需求。

6.1.1 EMS 对 NMS 消息的响应要有相关性

EMS 响应 NMS 的消息需要包含一个和 NMS 的输入命令相关的标识。在对 NMS 消息的回答中，EMS 要将 NMS 消息中的标识符包含在其中。NMS 负责在不同的 EMS 之间相关性标识符的惟一性。

6.1.2 EMS 自动上报给 NMS 的消息要包含序列号

对每个 EMS 上报给 NMS 的消息来说，EMS 都应当为这个消息分配一个序列号。这样的序列号应该包括在所有 EMS 自动上报的消息中。这些自动上报的消息可以是 EMS 产生的或者是 NE 产生的告警、事件以及数据库变化的报告。NMS 将利用这些序列号来区别遗失的 EMS 自动上报消息。

6.1.3 EMS 自动上报消息的日志

EMS 应当为所有 EMS 或者 NE 产生的自动上报消息提供日志记载功能，包括告警日志、性能日志和事件日志等。特殊的日志要求可以由协议相关的接口要求规定。

6.1.4 对 NMS/EMS 心跳的支持

EMS 应当向 NMS 发送周期性的心跳通知以表明 EMS 是否在用户定义的间隔期间保持活动状态。对心跳通知功能的禁止同样也要提供。

在 EMS 和 NMS 链路失败的情况下，网元的状态信息可能会改变，在链路恢复时，EMS 应该能够向 NMS 发出恢复的消息使得 NMS 可以据此来向 EMS 发出同步消息的请求，例如 NMS 可在收到 EMS 链路恢复的消息后向 EMS 发出同步告警的请求以获取链路失败的情况下丢失的告警信息。

6.1.5 获取 EMS 软件版本信息

NMS 应该能够要求 EMS 提供其软件版本、接口版本等相关软件系统信息。

6.1.6 网络拓扑发现

NMS 应该能够要求 EMS 提供网元管理系统中网络结构视图，应包括 EMS 中的逻辑分组、网元的信息以及它们之间的层次关系。NMS 应能够根据这些信息构建适时的物理网络拓扑图。

如果 EMS 能自动自主地发现网络拓扑，并在其数据库中存有最新的拓扑信息，它应根据 NMS 请求提供拓扑信息。

6.1.7 通知管理功能

通知管理功能可以进一步分解为：通知上报功能和通知订购功能集。通知订购功能集包括订购通知和撤销订购。

- 通知上报功能

与故障管理相关的通知：

1） 新的告警通知：表示一个新产生的告警（告警类型可为设备告警、环境告警、通信告警、处理错告警、QOS 告警等）。

2） 变化的告警通知（可选）：表示原有的一个告警其级别发生了变化。

3） 清除的告警通知（可选）：表示原有的一个告警已经被清除。

4） 告警确认状态改变通知（可选）： 见故障管理接口功能需求。

5） 增加告警说明通知（可选）： 见故障管理接口功能需求。

与心跳相关的通知：

心跳消息。

- 订购通知

该功能支持NMS通过北向接口订购相应通知，即NMS指定通知过滤条件及前向目的地，返回订购号，订购成功后，Agent将根据此过滤条件和目的地信息向NMS上报相应通知，NMS需要指定的信息包括：上报通知的目的地。

- 撤销订购

该功能支持 NMS 通过北向接口根据订购号撤销已经存在的通知订购。

6.2 配置管理接口功能需求

该小节描述DSL网络接口管理功能需求中的配置管理接口功能需求。本节所指的线路均为DSL线路。

6.2.1 获取网元列表

NMS 应该能够要求 EMS 提供它所管理的网元 ID 列表及明细列表。表中应包括如下信息：软件版本号，硬件版本号，序列号，MAC 地址，IP 地址，供应商及型号。

6.2.2 获取端口状态（ifOperstatus，ifAdminStatus）

NMS 应该能够要求 EMS 提供网元某个端口（给定端口 ID）的状态（运行状态、管理状态）。

6.2.3 启用/禁用端口

NMS 应该能够要求 EMS 启用/禁用（Enable/Disable）指定端口（给定端口 ID），将端口的管理状态指定为 Up/Down。

6.2.4 复位端口

NMS 应该能够要求 EMS 复位指定端口（给定端口 ID）。复位端口应该仅针对指定端口，如果不能做到针对指定端口，可以通过组合启用/禁用端口来模拟实现。

6.2.5 DSL 参数模板配置（创建、修改、删除）

NMS 应当能够通过 NMS/EMS 接口获取、创建以及指派 EMS 参数模板，NMS 也可以要求 EMS 将特定的参数模板应用到网元上（DSL 参数模板包括线路参数模板和告警参数模板）的假设在安装期间，当网元启动时，它会自动地使用缺省的参数模板，这个缺省的参数模板可以是设备出厂的参数模板或者其他。然而，如果需要一个不同的有别于缺省参数模板的参数模板时，NMS 应该能够要求 EMS 将 NMS 要求的参数模板应用到网元上，NMS 能够通过 NMS/EMS 间的命令要求 EMS 执行这个功能，所以，NMS 必须有权要求 EMS 存储参数模板并且将它应用到网元上。NMS 也可以要求对 EMS 对应用到网元上的参数模板进行更改。

不同的线路类型对应不同的线路参数模板，ADSL 线路对应 ADSL 线路参数模板，VDSL 线路对应 VDSL 线路参数模板，SHDSL 线路对应 SHDSL 线路参数模板。

不同的线路类型对应不同的告警参数模板，ADSL 线路对应 ADSL 告警参数模板，VDSL 线路对应 VDSL 告警参数模板，SHDSL 线路对应 SHDSL 告警参数模板。

6.2.6 获取线路配置信息

NMS 应该能够要求 EMS 获得指定 DSL 线路（给定线路 ID）的配置信息：ADSL 和 VDSL 线路包括线路类型、线路编码、线路参数模板和告警参数模板；ADSL2+线路包括线路参数模板、告警参数模板和传输模式；SHDSL 线路包括线路参数模板、告警参数模板和中继器数目。

6.2.7 线路配置

NMS 应该能够要求 EMS 对指定 DSL 线路（给定线路 ID）配置，针对不同的用户线技术，配置的内容不同：ADSL、VDSL 线路配置包括线路类型、线路编码和线路参数模板和告警参数模板；ADSL2+线路配置包括线路参数模板、告警参数模板和传输模式；SHDSL 线路配置包括线路参数模板、告警参数模板和中继器数目。

6.2.8 ADSL2+功耗模式配置

NMS 应该能够要求 EMS 对指定 ADSL2+线路（给定线路 ID）进行功耗模式配置，功耗模式有全功耗模式、低功耗模式和休眠功耗模式。

6.2.9 SHDSL 端点配置

NMS 应该能够要求 EMS 对指定 SHDSL 线路上的某个端点进行配置，配置内容包括告警参数模板。

6.3 性能管理接口功能需求

6.3.1 门限设置

NMS 应该能够要求 EMS 设置和修改网元的性能管理门限数据。

6.3.2 获取线路运行数据

NMS应能请求从EMS获取DSL线路运行数据，包括可能的时间戳（time stamp）。

6.3.3 获取线路性能设置数据

EMS 接口应该按线路提供性能相关的配置参数，比如线路的目标信噪比裕度以及最大、最小信噪比裕度。

6.3.4 获取线路性能监控数据

EMS 接口应该按线路提供性能相关的实时数据，如当前速率和信噪比裕度等。EMS 接口也应能按线路提供给 NMS 与性能相关的 24h 历史数据或 15min 历史数据。

6.4 故障管理接口功能需求

6.4.1 告警实时上报功能

该功能应支持 Agent 通过 Itf-N 接口实时上报告警信息。该功能通过公共管理功能中的“通知管理功能”的通知上报功能实现。告警实时性由被管系统保证，其实现方法不在本规范规定范围之内。

上报的告警类型包括：

— 新的告警：新产生的告警。

— 清除的告警：通知原有告警被清除的告警。

— 改变的告警：通知原有告警被改变的告警。

Manager 可通过通知管理接口订购告警时设置相关的告警上报过滤条件，可指定的过滤条件包括：

— 网元 ID；

— 网元类型；

— 告警产生时间；

— 告警级别；

— 告警原因。

6.4.2 同步告警信息功能

该功能是为了使Manager和Agent所拥有的告警信息保持一致性，被同步的告警为未被恢复的告警。

Manager应可指定同步过滤条件，可指定的同步过滤条件包括：

— 要同步的起始时间；

— 要同步的终止时间；

— 告警原因；

— 网元类型；

— 网元标识；

— 告警级别等。

在多Manager的情况下，若多个Manager同时发出同步请求，Agent应都支持。

Agent在返回告警同步信息时，应保证返回信息与Agent信息的一致性。

在告警信息同步过程中，新产生的告警应实时上报给网管系统。

6.4.3 告警确认功能（可选）

该功能支持Manager通过Itf-N接口对告警进行确认。一个Manager对某个告警进行了确认，此告警的确认状态由“未确认”变为“已确认”。

Agent应将相关的告警确认状态改变通知发送给所有订购了该通知的Manager。

Manager 对告警进行确认时应该提供以下信息：

— 被确认的告警相关信息：告警号。

— 进行告警确认操作的用户或系统的名称。

6.4.4 取消告警确认功能（可选）

该功能支持Manager通过Itf-N接口取消对某告警的确认，一个Manager对某个告警取消确认后，此告警的确认状态由“已确认”变为“未确认”。

Agent应将相应的告警确认状态改变通知发送给所有订购了该通知的Manager。

Manager 对告警进行取消确认时应该提供以下信息：

— 要取消确认的告警的告警号。

6.4.5 获取告警数功能（可选）

获取告警数功能应支持Manager通过Itf-N接口获取指定条件的告警数量。

Manager可输入要获取的告警数的条件，包括：

— 告警类型；

— 告警级别；

— 网元标识；

— 告警发生时间等。

6.4.6 增加告警说明功能（可选）

该功能应支持Manager通过Itf-N接口增加对某个或某类告警的补充说明（该补充说明可为Manager对某告警的处理经验，或针对某告警的提示信息、附加说明和注意事件等）。一个Manager对某告警增加了补充说明，Agent应将相关的“增加告警说明”通知发送给所有订购了该通知的Manager，以有利于多Manager环境下对告警信息的管理。

6.4.7 清除告警功能（可选）

该功能应支持Manager通过Itf-N接口主动清除告警。

在某些特定情况下，Manager可以主动清除指定告警，告警清除后，此告警由活跃告警变为已清除告警，即告警状态为“已清除”。一个Manager对某告警进行了清除后，Agent应将相关的告警清除通知实时发送给所有订购了该通知的Manager，采用谁清除谁负责的方式。

6.4.8 告警确认状态改变通知上报功能（可选）

Manager通过告警确认功能对某告警成功确认后，或者Agent对某告警成功确认后，Agent应将相关的告警确认状态改变通知发送给所有订购了该通知的Manager，采用谁确认谁负责的方式。该功能使用了上述公共管理接口功能的“通知管理功能”。

在多Manager的情况下，该通知为必选；在单Manager的情况下，该通知为可选。

6.4.9 增加说明通知上报功能（可选）

Agent完成“增加告警说明”操作后发送该通知到所有订购了该通知的Manager。

该功能使用了上述公共管理接口功能的“通知管理功能”。

在多Manager的情况下，该通知为必选；在单Manager的情况下，该通知为可选。

6.5 安全管理接口功能需求（可选）

6.5.1 用户组管理

NMS 应该能够要求通过 EMS 对用户组管理，给定用户组 ID，以及为该用户组指定的操作权限范围去创建一个用户组。给定一个已经存在的用户组 ID 要求删除这个用户组，还可以对给定的用户组修改其操作权限。对用户组的删除和修改要相应改变原先属于该用户组的用户信息。

6.5.2 用户管理

用户的管理是基于用户组的。NMS 应该能够要求通过 EMS 对用户进行管理。给定用户 ID，用户口令，为用户指定的组 ID，能够在 EMS 上创建一个用户。可以为已存在的用户更改口令，可以为已存在的用户重新指定所属的组 ID，可以删除已有的用户，可以为用户指定相应的管理域。

6.5.3 管理域管理

NMS 应该能够要求通过 EMS 对管理域进行管理，能够在 EMS 上通过给定管理域 ID 和管理域管理的范围（资源 ID 列表）来创建管理域，能够删除该管理域。对管理域的删除和修改会对已分配该管理域的用户相应改变。

6.5.4 操作权限管理

NMS 应该能够要求通过 EMS 对操作权限进行管理，能够为用户组分配权限。对特定用户来说，对其管理的资源只能施加其权限范围内的操作，保护资源不受到不合法的访问。

6.5.5 EMS-NMS 认证鉴权管理

EMS 和 NMS 之间必须提供认证鉴权管理，根据提供的用户 ID 和口令来获得 EMS 相应类型的服务。

7 信息模型

7.1 对象管理实体

7.1.1 EMS

EMS 代表逻辑上的网络管理系统，它管理了一种或多种类型的网元。

关键属性：

EMS IP 地址

EMS 版本号

关键操作：

获取 EMS 管理的所有网元列表

获取 EMS 相关的所有当前告警

7.1.2 分组（Location）

代表网元管理系统中的逻辑分组，该分组中可以包含多个逻辑的管理单元。

关键属性：

分组标识

分组名称

父分组标识

关键操作：

获取指定分组的所有网元信息列表

7.1.3 网元（NE）

代表一个逻辑的管理单元，它可能包含多个协同工作的设备。这些协同工作的设备组合起来作为 NMS

可以管理的一个逻辑单元。

关键属性：

网元标识

网元类型

网元名称

网元版本

网元带内 MAC 地址

网元带外 MAC 地址

关键操作：

获取网元包含的功能设备列表

获取该网元所支持的 ADSL 线路

7.1.4 Card（板卡）

板卡是网元中的一个物理功能设备。可以被安装或者替换，也就是可以在槽位中插入或移除。

关键属性：

板卡标识

板卡类型

板卡类型描述

板卡软件版本

板卡硬件版本

板卡上端口数

关键操作：

获取板卡信息

7.2 ADSL 传输管理实体

7.2.1 ADSL 线路

ADSL 线路描述 ADSL 传输设备，包括 ATU-C 和 ATU-R。

关键属性：

ADSL 线路 ID：惟一标识 ADSL 线路，只读

管理状态（Administrative State）：读写

运行状态（Operational State）：只读

用户标识（User Label）：读写

ADSL 参数配置模板（ADSL Line Configuration Profile）：读写

ADSL 告警参数模板（ADSL Alarm Configuration Profile）：读写

线路编码（Line Coding）：读写

线路类型（Line Type）：读写

当前信噪比裕度（Current SNR Margin）：只读

当前衰减（Current Attenuation）：只读

当前输出功率（Current Output Power）：只读

当前可达速率（Current Attainable Rate）：只读

关键操作：

ADSL 线路配置

7.2.2 ADSL2+线路

ADSL2+线路描述 ADSL2+传输，包括 ATU-C 和 ATU-R。

关键属性：

ADSL 线路 ID：惟一标识 ADSL 线路，只读

管理状态（Administrative State）：读写

运行状态（Operational State）：只读

用户标识（User Label）：读写

ADSL 线路参数模板（ADSL Configuration Profile）：读写

ADSL 告警参数模板（ADSL Aalarm Configuration Profile）：读写

传输模式(Trans Mode)：读写

当前信噪比裕度（Current SNR Margin）：只读

当前衰减（Current Attenuation）：只读

当前输出功率（Current Output Power）：只读

当前可达速率（Current Attainable Rate）：只读

关键操作：

ADS2+线路配置

7.2.3 ADSL 信道

ADSL 信道是由 ADSL 线路承载的信道。有两种类型的信道被定义，一个是快速信道，另一个是交织信道。

关键属性：

ADSL 信道 ID（ADSL Channel ID）：惟一标识这个 ADSL 信道，只读

管理状态（Administrative State）：读写

运行状态（Operational State）：只读

信道类型（Channel Type）：只读

当前信道速率（Current Channel Rate）：包括 ATU-C 和 ATU-R，只读

当前信道交织延迟（Current Channel Interleave Delay）：只用于交织信道，包括 ATU-C 和 ATU-R，只读

当前循环冗余检验块长度（Current CRC Block Length）：包括 ATU-C 和 ATU-R，只读

当前线速率（Current Line Rate）：包括 ATU-C 和 ATU-R，只读 (删除)

关键操作：

获得相关 ADSL 线路实体

7.2.4 ADSL 线路参数模板

ADSL 线路参数模板是 ADSL 线路和 ADSL 信道的配置参数的集合，包括 ATU-C 和 ATU-R 两端。

关键属性：

线路参数模板名称 (Profile Name)：惟一标识这个线路参数模板，读写

速率适应模型(Rate Adaptation Mode)：包括 ATU-C 和 ATU-R，读写

速率通道比率(Rate Channel Ratio)：包括 ATU-C 和 ATU-R，读写

最小信噪比裕度(Min SNR Margin)：包括 ATU-C 和 ATU-R，读写

最大信噪比裕度(Max SNR Margin)：包括 ATU-C 和 ATU-R，读写

目标信噪比裕度(Target SNR Margin)：包括 ATU-C 和 ATU-R，读写

最小快速信道速率(Min Fast Channel Rate)：包括 ATU-C 和 ATU-R，读写

最大快速信道速率(Max Fast Channel Rate)：包括 ATU-C 和 ATU-R，读写

最小交织信道速率(Min Interleaved Channel Rate)：包括 ATU-C 和 ATU-R，读写

最大交织信道速率(Max Interleaved Channel Rate)：包括 ATU-C 和 ATU-R，读写

最大交织信道延迟(Max Interleaved Channel Delay)：包括 ATU-C 和 ATU-R，读写

关键操作：

ADSL 线路参数模板的创建、修改和删除

7.2.5 ADSL2+线路参数模板

ADSL2+线路参数模板是ADSL2+线路和ADSL2+信道的配置参数的集合，包括ATU-C和ATU-R两端，除了含有ADSL线路参数模板中的所有内容以外，还含有线路类型。

关键属性：

ADSL2线路参数模板内容

线路类型

关键操作：

ADSL2+线路参数模板的创建、修改和删除

7.2.6 ADSL 告警参数模板

ADSL 告警配置模板是 ADSL 线路和 ADSL 信道的告警参数的集合，包括 ATU-C 和 ATU-R 两端。

关键属性：

告警参数模板名称（Profile Name）：惟一标识这个 ADSL 告警参数模板，只读

15min 最小帧丢失门限（Threshold for Loss of Frame）：包括 ATU-C 和 ATU-R，读写

15min 最小信号丢失门限（Threshold for Loss of Signal）：包括 ATU-C 和 ATU-R，读写

15min 最小链路丢失门限（Threshold for Loss of Link） ：只包括 ATU-C，读写

15min 最小功率丢失门限（Threshold for Loss of Power）：包括 ATU-C 和 ATU-R，读写

15min 最小错误秒门限（Threshold for Error Second）：包括 ATU-C 和 ATU-R，读写

快速信道数率上升门限（Threshold for Fast Channel Rate Change Up ）：包括 ATU-C 和 ATU-R，读写

交织信道数据上升门限（Threshold for Interleave Channel Rate Change Up）：包括 ATU-C 和 ATU-R，读写

快速信道数率下降门限（Threshold for Fast Channel Rate Change Down）：包括 ATU-C 和 ATU-R，读写

交织信道数据下降门限（Threshold for Interleave Channel Rate Change Down）：包括 ATU-C 和 ATU-R，读写

使能初始化失败告警（Initialize Failure Trap）：只包括 ATU-C，读写

关键操作：

获得相关的 ADSL 门限

7.2.7 ADSL2+告警参数模板

ADSL2+告警配置模板是ADSL2+线路和ADSL2+信道的告警参数的集合，包括ATU-C和ATU-R两端。

关键属性：

ADSL告警参数模板内容

Atuc快速训练失败告警门限

Atuc线路严重误码告警门限

Atuc线路不可用告警门限

Atur线路严重误码告警门限

Atur线路不可用告警门限

7.3 ADSL 性能监控实体

7.3.1 ADSL 实时性能监控模型(C 端)

ADSL 实时性能监控模型代表 ADSL 信道上 C 端的实时性能参数模型。

关键属性：

衰减（Line Attenuation）

噪声裕度（Noise Margin）

输出功率（Total Output Power）

最大可达速率（Maximum Attainable Rate）

当前速率（Current Rate）

前一次速率（Previous Rate）

Channel Data Block Length

交织延迟（Interleave Delay）

信号丢失数(Loss of Signal)

帧丢失数（Loss of Frame）

功率丢失数（Loss of Power）

链路丢失数（Loss of Link）

错误秒数（Errored Seconds）

发送数据块数（Transmit Blocks）

接收数据块数（Receive Blocks）

纠正数据块数（Corrected Blocks）

未纠正数据块数（Uncorrectable Blocks）

关键操作：

读取一条线路上 C 端的实时性能参数

7.3.2 ADSL 实时性能监控模型（R 端）

ADSL 实时性能监控模型代表 ADSL 信道上 R 端的实时性能参数模型。

关键属性：

衰减（Line Attenuation）

噪声裕度（Noise Margin）

输出功率（Total Output Power）

最大可达速率（Maximum Attainable Rate）

当前速率（Current Rate）

前一次速率（Previous Rate）

Channel Data Block Length

交织延迟（Interleave Delay）

信号丢失数(Loss of Signal)

帧丢失数（Loss of Frame）

功率丢失数（Loss of Power）

链路丢失数（Loss of Link）

错误秒数（Errored Seconds）

发送数据块数（Transmit Blocks）

接收数据块数（Receive Blocks）

纠正数据块数（Corrected Blocks）

未纠正数据块数（Uncorrectable Blocks）

关键操作：

读取一条线路上 R 端的实时性能参数

7.3.3 ADSL15min 性能统计模型（C 端）

ADSL15min 性能统计模型代表 ADSL 信道上 C 端 15min 内的性能数据统计模型。

关键属性：

信号丢失数(Loss of Signal)

帧丢失数（Loss of Frame）

功率丢失数（Loss of Power）

链路丢失数（Loss of Link）

错误秒数（Errored Seconds）

发送数据块数（Transmit Blocks）

接收数据块数（Receive Blocks）

纠正数据块数（Corrected Blocks）

未纠正数据块数（Uncorrectable Blocks）

关键操作：

读取一条线路上 C 端的 15min 性能统计数据

7.3.4 ADSL15min 性能监控模型（R 端）

ADSL15min 性能统计模型代表 ADSL 信道上 R 端 15min 内的性能数据统计模型。

关键属性：

信号丢失数(Loss of Signal)

帧丢失数（Loss of Frame）

功率丢失数（Loss of Power）

链路丢失数（Loss of Link）

错误秒数（Errored Seconds）

发送数据块数（Transmit Blocks）

接收数据块数（Receive Blocks）

纠正数据块数（Corrected Blocks）

未纠正数据块数（Uncorrectable Blocks）

关键操作：

读取一条线路上 R 端的 15min 性能统计数据

7.3.5 ADSL 当前一天性能监控模型（C 端）

ADSL 当前一天性能统计模型代表 ADSL 信道上 C 端当前一天内的性能数据统计模型。

关键属性：

信号丢失数(Loss of Signal)

帧丢失数（Loss of Frame）

功率丢失数（Loss of Power）

链路丢失数（Loss of Link）

错误秒数（Errored Seconds）

发送数据块数（Transmit Blocks）

接收数据块数（Receive Blocks）

纠正数据块数（Corrected Blocks）

未纠正数据块数（Uncorrectable Blocks）

关键操作：

读取一条线路上 C 端的当前一天性能统计数据

7.3.6 ADSL 当前一天性能监控模型（R 端）

ADSL 当前一天性能统计模型代表 ADSL 信道上 R 端当前一天内的性能数据统计模型。

关键属性：

信号丢失数(Loss of Signal)

帧丢失数（Loss of Frame）

功率丢失数（Loss of Power）

链路丢失数（Loss of Link）

错误秒数（Errored Seconds）

发送数据块数（Transmit Blocks）

接收数据块数（Receive Blocks）

纠正数据块数（Corrected Blocks）

未纠正数据块数（Uncorrectable Blocks）

关键操作：

读取一条线路上 R 端的当前一天性能统计数据

7.3.7 ADSL 前一天性能监控模型（C 端）

ADSL 当前一天性能统计模型代表 ADSL 信道上 R 端前一天内的性能数据统计模型。

关键属性：

信号丢失数(Loss of Signal)

帧丢失数（Loss of Frame）

功率丢失数（Loss of Power）

链路丢失数（Loss of Link）

错误秒数（Errored Seconds）

发送数据块数（Transmit Blocks）

接收数据块数（Receive Blocks）

纠正数据块数（Corrected Blocks）

未纠正数据块数（Uncorrectable Blocks）

关键操作：

读取一条线路上 C 端的前一天性能统计数据

7.3.8 ADSL 前一天性能监控模型（R 端）

ADSL 当前一天性能统计模型代表 ADSL 信道上 R 端前一天内的性能数据统计模型。

关键属性：

信号丢失数(Loss of Signal)

帧丢失数（Loss of Frame）

功率丢失数（Loss of Power）

链路丢失数（Loss of Link）

错误秒数（Errored Seconds）

发送数据块数（Transmit Blocks）

接收数据块数（Receive Blocks）

纠正数据块数（Corrected Blocks）

未纠正数据块数（Uncorrectable Blocks）

关键操作：

读取一条线路上 R 端的前一天性能统计数据

7.4 VDSL 传输管理实体

7.4.1 VDSL 线路

VDSL 线路描述 VDSL 传输设备，包括 VTUC 和 VTUR。

关键属性：

VDSL 线路 ID：惟一标识 VDSL 线路，只读

管理状态（Administrative State）：读写

运行状态（Operational State）：只读

用户标识（User Label）：读写

线路编码类型（Line Coding Type）：只读

线路类型（Line Type）：只读

VDSL 线路参数模板（VDSL Line Profile）：读写

VDSL 告警参数模板（VDSL Line Profile）：读写

关键操作：

获得相关的 VDSL 信道

获得相关的设备实体

7.4.2 VDSL 信道

VDSL 信道是由 ADSL 线路承载的信道。有两种类型的信道被定义，一个是快速信道，一个是交织信道。

关键属性：

VDSL 信道 ID（VDSL Channel ID）：惟一标识这个 VDSL 信道，只读

信道交织延迟（Channel Interleave Delay）：只用于交织信道，只读

循环冗余检验块长度（CRC Block Length）：只读

当前信速速率（Current Channel Rate）：只读

关键操作：

获得相关 VDSL 线路实体

7.4.3 VDSL 线路参数模板

VDSL 线路参数模板是 VDSL 线路配置参数的集合。

关键属性：

线路参数模板 ID(Profile ID)： 惟一标识这个 VDSL 线路参数模板，只读

线路参数模板名称 (Profile Name)： 惟一标识这个线路参数模板，读写

速率模式(Rate Mode)：包括上行和下行，读写

最大功率(Max Power)：包括上行和下行，读写

最大信噪比裕度(Max SNR Margin)：包括上行和下行，读写

最小信噪音比裕度(Min SNR Margin)：包括上行和下行，读写

目标信噪音比裕度(Target SNR Margin)：包括上行和下行，读写

最小快速信道速率(Min Fast Channel Rate)：包括上行和下行，读写

最大快速信道速率(Max Fast Channel Rate)：包括上行和下行，读写

最小交织信道速率(Min Interleaved Channel Rate)：包括上行和下行，读写

最大交织信道速率(Max Interleaved Channel Rate)：包括上行和下行，读写

最大交织信道延迟(Max Interleaved Channel Delay)：包括上行和下行，读写

PBO 使能(Power Back-off Control)：包括上行和下行，读写

关键操作：

获取、添加、修改和删除 VDSL 线路参数模板

7.4.4 VDSL 告警参数模板

VDSL 告警参数模板是 VDSL 线路的告警参数模板集合。

关键属性：

告警参数模板名称（Profile Name）：惟一标识这个 ADSL 告警参数模板，只读

15min 最小帧丢失秒门限（Threshold for Loss of Frame Seconds）：读写

15min 最小信号丢失秒门限（Threshold for Loss of Signal Seconds）：读写

15min 最小电源丢失秒门限（Threshold for Loss of Power Seconds）：读写

15min 最小链路丢失秒门限（Threshold for Loss of Link Seconds）：读写

15min 最小错误秒门限（Threshold for Error Seconds）：读写

15min 最小严重错误秒门限（Threshold for Severely Error Seconds）：读写

15min 最小不可达秒门限（Threshold for Unavailable Seconds）：读写

使能初始化失败告警（Initialize Failure Trap）：读写

关键操作：

获取、添加、修改和删除 VDSL 告警参数模板

获得相关的 VDSL 告警门限

7.5 VDSL 性能监控实体

7.5.1 VDSL 实时性能监控模型

VDSL 实时性能监控模型代表 VDSL 信道上的实时性能参数模型。

关键属性：

帧丢失秒数（Loss of Frame Seconds）

信号丢失秒数(Loss of Signal Seconds)

电源丢失秒数（Loss of Power Seconds）

链路丢失秒数（Loss of Link Seconds）

错误秒数（Errored Seconds）

严重错误秒数（Severely Errored Seconds）

不可达秒数（Unavailable Seconds）

纠正字节数（Corrected Octets）

未纠正数据块数（Uncorrectable Blocks）

关键操作：

读取一条线路上的实时性能参数

7.5.2 VDSL15min 性能统计模型

VDSL15min 性能统计模型代表 VDSL 信道上 15min 内的性能数据统计模型。

关键属性：

帧丢失秒数（Loss of Frame Seconds）

信号丢失秒数(Loss of Signal Seconds)

电源丢失秒数（Loss of Power Seconds）

链路丢失秒数（Loss of Link Seconds）

错误秒数（Errored Seconds）

严重错误秒数（Severely Errored Seconds）

不可达秒数（Unavailable Seconds）

纠正字节数（Corrected Octets）

未纠正数据块数（Uncorrectable Blocks）

关键操作：

读取一条线路上的 15min 性能统计数据

7.5.3 VDSL 当前一天性能监控模型

VDSL 当前一天性能统计模型代表 VDSL 信道上当前一天内的性能数据统计模型。

关键属性：

帧丢失秒数（Loss of Frame Seconds）

信号丢失秒数(Loss of Signal Seconds)

电源丢失秒数（Loss of Power Seconds）

链路丢失秒数（Loss of Link Seconds）

错误秒数（Errored Seconds）

严重错误秒数（Severely Errored Seconds）

不可达秒数（Unavailable Seconds）

纠正字节数（Corrected Octets）

未纠正数据块数（Uncorrectable Blocks）

关键操作：

读取一条线路上的当前一天性能统计数据

7.6 SHDSL 传输管理实体

7.6.1 SHDSL 线路

SHDSL 线路描述 SHDSL 传输设备，包括 STUC 和 STUR。

关键属性：

SHDSL 线路 ID：惟一标识 SHDSL 线路，只读

管理状态（Administrative State）：读写

运行状态（Operational State）：只读

用户标识（User Label）：读写

SHDSL 线路参数模板（SHDSL Line Profile）：读写

SHDSL 告警参数模板（SHDSL Alarm Profile）：读写

中继器数目(Repeater Number)：读写

关键操作：

获取和设置相关的 SHDSL 线路参数

获得相关的设备实体

7.6.2 SHDSL 端点

SHDSL 端点描述 SHDSL 传输链路上单元设备的用户侧或网络侧。

关键属性：

SHDSL 线路 ID：惟一标识 SHDSL 线路，只读

端点（Endpoint）：只读

端点侧（Endpoint Side）：只读

端点线对（Endpoint Wire Pair）：只读

端点告警参数模板（Endpoint Alarm Profile）：读写

关键操作：

获取和设置相关的 SHDSL 端点参数

7.6.3 SHDSL 线路参数模板

SHDSL 线路参数模板是 SHDSL 线路配置参数的集合。

关键属性：

线路参数模板 ID（Profile ID）：惟一标识这个 SHDSL 线路参数模板，只读

线路参数模板名称（Profile Name）：惟一标识这个线路参数模板，读写

线路速率（Line Rate）：包括最大值和最小值，单位为 kbit/s，读写

传输模式（Transmission Mode）：读写

线制（Wire Interface）：读写

PSD（PSD）：读写

线路 Probe（Line Probe）：读写

时钟参考类型（Clock Reference Type）：读写

电源馈电（Power Feeding）：读写

远程管理（Remote Management）：读写

STUC 当前条件下目标信噪比裕度（STUC Current Condition Target SNR Margin）：单位为 db，读写

STUC 最坏情况下目标信噪比裕度（STUC Worst Case Target SNR Margin）：单位为 db，读写

STUR 当前条件下目标信噪比裕度（STUR Current Condition Target SNR Margin）：单位为 db，读写

STUR 最坏情况下目标信噪比裕度（STUR Worst Case Target SNR Margin）：单位为 db，读写

关键操作：

获取、添加、修改和删除 SHDSL 线路参数模板

7.6.4 SHDSL 告警参数模板

SHDSL 告警参数模板是 SHDSL 线路告警参数的集合。

关键属性：

告警参数模板名称（Profile Name）：惟一标识这个 SHDSL 告警参数模板，只读

环路衰减告警门限（Loop Attenuation Threshold）：单位为 db，读写

信噪比裕度门限（SNR Margin Threshold）：单位为 db，读写

15min 内误码秒门限（ES Within 15Min Threshold）：单位为 second，读写

15min 内严重误码秒门限（SES Within 15Min Threshold）：单位为 second，读写

15min 内 CRC 异常计数门限（CRC Anomalies Within 15Min Threshold）：读写

15min 内同步字丢失秒数门限（LOSWS Within 15Min Threshold）：单位为 second，读写

15min 内不可用秒数门限（UAS Within 15Min Threshold）：单位为 second，读写

关键操作：

获取、添加、修改和删除 SHDSL 告警参数模板

获得相关的 SHDSL 告警门限

7.7 SHDSL 性能监控实体

7.7.1 SHDSL 端点实时性能监控模型

SHDSL 实时性能监控模型代表 SHDSL 端点上的实时性能参数模型。

关键属性：

端点当前环路衰减（EndpointCurrAtn）：只读

端点当前信噪比裕度（EndpointCurrSnrMgn）：只读

端点发生错误的秒数（EndpointES）：只读

端点发生严重错误的秒数（EndpointSES）：只读

端点 CRC 异常计数（EndpointCRCanomalies）：只读

端点错误同步字秒数（EndpointLOSWS）：只读

端点难以获得秒数（EndpointUAS）：只读

关键操作：

读取某个 SHDSL 端点的实时性能参数

7.7.2 SHDSL 端点 15min 性能监控模型

SHDSL 实时性能监控模型代表 SHDSL 端点上的 15min 内的性能参数模型。

关键属性：

端点当前 15min 内过去的秒数（Endpoint Curr15MinTimeElapsed）：只读

端点当前 15min 内发生错误的秒数（Endpoint Curr15MinES）：只读

端点当前 15min 内发生严重错误的秒数（Endpoint Curr15MinSES）：只读

端点当前 15min 内 CRC 异常计数（Endpoint Curr15MinCRCanomalies）：只读

端点当前 15min 内错误同步字秒数（Endpoint Curr15MinLOSWS）：只读

端点当前 15min 内难以获得秒数（Endpoint Curr15MinUAS）：只读

关键操作：

读取某个 SHDSL 端点的 15min 内的性能参数

7.7.3 SHDSL 端点当前一天性能监控模型

SHDSL 实时性能监控模型代表 SHDSL 端点上的当前一天内的性能参数模型。

关键属性：

端点当前 1 天内过去的秒数（Endpoint Curr1DayTimeElapsed）：只读

端点当前当前一天内发生错误的秒数（Endpoint Curr1DayES）：只读

端点当前当前一天内发生严重错误的秒数（Endpoint Curr1DaySES）：只读

端点当前当前一天内 CRC 异常计数（Endpoint Curr1DayCRCanomalies）：只读

端点当前当前一天内错误同步字秒数（Endpoint Curr1DayLOSWS）：只读

端点当前当前一天内难以获得秒数（Endpoint Curr1DayUAS）：只读

关键操作：

读取某个 SHDSL 端点的当前一天内的性能参数

7.8 故障管理实体

故障管理实体定义的是上述故障管理接口功能要求提到的告警实时上报中的通知参数到通知结构的映射。

通知的组成有通知头和通知体，通知头由域名、类型名和事件名组成。通知体由数组构成，数组中的每个元素都是一个名值对，名字表示属性的含义，值表示属性的取值。不同的告警上报通知类型具有不同的通知体，详细的参数映射关系详见 8.3.2 描述。

7.8.1 NotifyNewAlarm 的通知参数映射

描述上报事件类型名为 NotifyNewAlarm 的事件的参数映射。

关键属性：

域名：字符串，对告警上报来说为“Alarm”

类型名：字符串常量， NotificationType::NOTIFY_FM_NEW_ALARM

事件名：字符串，由alarmconstdef模块中的AlarmType接口义。其值为下面取值之一：通信告警communications alarm，运行错误告警processing error alarm，环境告警environmental alarm，服务质量告警quality of service alarm，设备告警equipment alarm

可变事件头：无

告警源：告警产生的资源 ID，可过滤条件

告警源类型：可过滤条件

告警原因：可过滤条件

告警级别：可过滤条件

告警产生的时间：可过滤条件，指明事件发生时间。使用 OMG 定义的 UTC 类型

实体标识：用来可以惟一标识一个告警的实体，以便清除告警时确认清除的是何条告警，不可过滤条件

告警标识：不可过滤条件

关键操作：

无

7.8.2 notifyComments 的通知参数映射

描述上报事件类型名为 NOTIFY_FM_COMMENT_ADDED 的事件的参数映射。

关键属性：

域名：字符串，对告警上报来说为“Alarm”

类型名：字符串常量，alarmconstdef:: NotificationType::NOTIFY_FM_NEW_ALARM

事件名：字符串，由alarmconstdef模块中的AlarmType接口义。其值为下面取值之一：通信告警communications alarm，运行错误告警processing error alarm，环境告警environmental alarm，服务质量告警quality of service alarm，设备告警equipment alarm

可变事件头：无

告警源：告警产生的资源 ID，可过滤条件

告警源类型：可过滤条件

告警原因：可过滤条件

告警级别：可过滤条件

告警发生的时间：可过滤条件，指明事件发生时间。使用 OMG 定义的 UTC 类型

实体标识：用来可以惟一标识一个告警的实体，以便清除告警时确认清除的是何条告警，不可过滤条件

告警标识：不可过滤条件

告警注释：不可过滤条件

关键操作：

无

7.8.3 notifyAckStateChanged 的通知参数映射

描述上报事件类型名为 NOTIFY_FM_ ACK_STATE_CHANGED 的事件的参数映射。

关键属性：

域名：字符串，对告警上报来说为“Alarm”

类型名：字符串常量，alarmconstdef:: NotificationType::NOTIFY_FM_NEW_ALARM

事件名：字符串，由alarmconstdef模块中的AlarmType接口义。其值为下面取值之一：通信告警communications alarm，运行错误告警processing error alarm，环境告警environmental alarm，服务质量告警quality of service alarm，设备告警equipment alarm

可变事件头：无

告警源：告警产生的资源 ID，可过滤条件

告警源类型：可过滤条件

告警原因：可过滤条件

告警级别：可过滤条件

告警发生的时间：可过滤条件，指明事件发生时间。使用 OMG 定义的 UTC 类型。

实体标识：用来可以惟一标识一个告警的实体，以便清除告警时确认清除的是何条告警，不可过滤条件

告警标识：不可过滤条件

确认人标识：不可过滤条件

确认系统标识：不可过滤条件

确认状态：不可过滤条件

关键操作：

无

7.8.4 notifyClearedAlarm 的通知参数映射

描述上报事件类型名为 NOTIFY_FM_ CLEARED_ALARM 的事件的参数映射

关键属性：

域名：字符串，对告警上报来说为“Alarm”

类型名：字符串常量，alarmconstdef:: NotificationType::NOTIFY_FM_NEW_ALARM

事件名：字符串，由alarmconstdef模块中的AlarmType接口义。其值为下面取值之一：通信告警communications alarm，运行错误告警processing error alarm，环境告警environmental alarm，服务质量告警quality of service alarm，设备告警equipment alarm

可变事件头：无

告警源：告警产生的资源 ID，可过滤条件

告警源类型：可过滤条件

告警原因：可过滤条件

告警级别：可过滤条件

告警发生的时间：可过滤条件，指明事件发生时间。使用 OMG 定义的 UTC 类型。

实体标识：用来可以惟一标识一个告警的实体，以便清除告警时确认清除的是何条告警，不可过滤条件

告警标识：不可过滤条件

清除人：不可过滤条件

清除系统：不可过滤条件

关键操作：

无

7.9 安全管理实体（可选）

安全管理实体主要用来描述 NMS 通过 NMS-EMS 接口对 EMS 的用户组、用户等安全信息进行管理时的信息格式。通过对安全管理实体的描述，可以通过鉴权认证机制控制对资源的操作，控制对 EMS 的操作。

7.9.1 用户组模型

用户组模型主要是定义一个操作权限集合，使得分配于其中的用户拥有这些操作权限。可以根据需要对权限进行划分，创建不同管理权限的用户组。方便属于其中的用户对资源相应操作特性的管理。

关键属性：

用户组的惟一标识

用户组关联的操作权限列表（权限 ID 列表）

关键操作：

创建用户组

删除用户组：相应对原属于该用户组的用户做相应的修改

为用户组分配权限

7.9.2 用户模型

用户的管理是基于用户组的，一个用户可以属于不只一个的用户组合，它所拥有的权限就是这些用户组操作权限的并集。还可以为用户指定管理域，这样就使得该用户只对他所管辖范围内的资源拥有定义的操作权限。可以为用户指定不只一个的管理域，这样该用户的管辖范围就是所有这些管理域包含的所有资源。

关键属性：

用户的惟一标识

用户所属的组 ID 列表

用户的口令

为用户指定的管理域 ID

关键操作：

创建用户

修改用户密码

删除用户

指定用户的管理域

指定用户的所属组

7.9.3 管理域模型

管理域就是对可管辖的资源 ID 进行划分，NMS 可以根据自己的实际需要划分管理范围，并且为这些管理范围建立相应的管理域，这样方便用户对特定资源的管理。

关键属性：

管理域 ID

管理域范围：可管辖的资源 ID 列表

关键操作：

创建管理域

删除管理域

指定管理域管理的资源 ID 列表

7.9.4 权限树模型

权限树就是资源上支持的操作特性的划分，NMS 可以根据自己的实际需要规划操作特性，对某些操作特性进行分类，例如，可将 EMS 管理系统中的操作特性分为安全管理权限、配置管理权限、故障管理权限和性能管理权限等。方便为不同用户指派操作权限范围。

关键属性：

权限惟一标识

权限关联的操作特性

关键操作：

添加权限

为权限指派操作特性

删除权限

8 接口操作定义

本节提供了NMS-EMS管理接口的细节描述，完整地提供了接口方法的相关参数和返回值，在本小节中使用sequence of 指一组同类型对象的序列，它是多个对象的集合。

8.1 公共管理

公共管理中包含对象管理和通知管理。

8.1.1 对象管理

对象管理主要是通用对象管理——NMS从EMS获得全网的分组信息，以及DSL设备管理——NMS从EMS获得设备的相关资源信息。

8.1.1.1 接口操作表

对象管理拓扑层次如图6所示。

表1　对象管理接口操作

接口名	描　述
getEmssys	查询EMS信息
getTopoTree	查询全网分组以及网元拓扑信息
getNeTopo	查询指定网元的资源信息

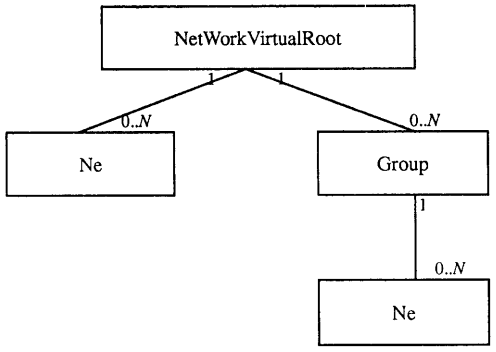


图 6　对象管理拓扑层次

8.1.1.2　接口操作参数和结果说明

表 2　getEmssys 接口操作参数

输入参数：无

输出/返回：

参　数	说　明
result	EmsSysInfo，Ems系统信息

表 3　EmsSysInfo 组成

属　性	名　称	描　述	类　型	限定符
emsIp	EMS IP	EMS IP地址，例如“192.10.1.28”	string	M,R
emsVer	EMS版本号	EMS 软件版本号，规则是厂商英文缩写+空格+EMS发布版本号，例如“HUAWEI iManager N2000 BMS V200R007B02D051” 各厂商缩写如下： 中兴：ZTE 华为：HUAWEI 阿尔卡特：ALCATEL UTStarco：UTSTARCOM 港湾：HARBOUR 西门子：SIEMENS 爱立信：ERICSSON 斯威特：SVT 其他：OTHERS	string	M,R

表 4　getTopoTree 接口操作参数

输入参数：无

输出/返回：

参　数	说　明
result	sequence of MO,所有分组以及网元信息列表，采用中序形式返回

表 5 MO 组成

属 性	名 称	描 述	类 型	限定符
category	管理对象类别	有：NetWorkVirtualRoot，Ne，Group 3种	string	M,R
attributes	管理对象的相关属性	MOAttribute由name，value组成，类型均为string	sequence of MOAttribute	M,R

表 6 NetWorkVirtualRoot 对象信息组成

NetWorkVirtualRoot 用来作为拓扑树的根节点。

属 性	名 称	描 述	类 型	限定符
ID	对象标识		string	M,R
NAME	对象名		string	M,R
PARENT	父对象标识		string	M,R
ISLEAF	是否是叶节点		boolean	M,R

表 7 Group 对象信息组成

Group 表示拓扑树的逻辑分组或者区域。

属 性	名 称	描 述	类 型	限定符
ID	对象标识		string	M,R
NAME	对象名		string	M,R
PARENT	父对象标识		string	M,R
ISLEAF	是否是叶节点		boolean	M,R

表 8 Ne 对象信息组成

属 性	名 称	描 述	类 型	限定符
ID	对象标识		string	M,R
NAME	对象名		string	M,R
PARENT	父对象标识		string	M,R
ISLEAF	是否是叶节点		boolean	M,R
NEIPADDR	设备 IP 地址		string	M,R
NETYPE	设备类型		string	M,R
DN	设备号		string	O,R

表 9 getNeTopo 接口操作参数

输入参数：

参 数	说 明
neips	sequence of string，设备IP地址列表

输出/返回：

参 数	说 明
result	sequence of NeInfo，网元信息列表

表 10 NeInfo 组成

属 性	名 称	描 述	类 型	限定符
neid	网元标识	EMS 所识别的网元的惟一标识	string	M,R
inbandmac	带内 MAC 地址		string	M,R
outbandmac	带外 MAC 地址		string	M,R
netype	网元类型	设备的型号参数	string	M,R
label	网元名称		string	M,R
version	网元版本	设备当前使用中的软件版本	string	M,R
vendorname	设备厂商名称		string	M,R
locationinfo	网元		string	O,R
slotnum	网元所含槽位数		short	O,R
cardInfo	网元所含的卡信息列表		sequence of CardInfo	M,R

表 11 CardInfo 组成

属 性	名 称	描 述	类 型	限定符
cardid	板卡标识	板卡所在的网元能够所识别的网元的惟一标识	short	M,R
cardtpye	板卡类型	板卡的型号参数	enum CardTypeDef{ CARD_WANIP, CARD_WANATM, CARD_ADSL, CARD_SHDSL, CARD_VDSL, CARD_IMA, CARD_E1, CARD_LAN, CARD_CORE, CARD_OTHER };	M,R
cardtypedesc	板卡类型描述		string	O,R
portnum	板卡上端口数		short	M,R
hardwareversion	板卡硬件版本		string	M,R
softwareversion	板卡软件版本		string	M,R
cpuload	cpu 负载	cpu 的使用率	short	O,R
memusage	内存占用	内存占用率	short	O,R

8.1.2 通知管理

8.1.2.1 接口操作表

表 11 通知管理接口操作

接口名	描 述
notreg	订购通知
disNotReg	取消订购

8.1.2.2 接口操作参数和结果说明

表 12 notreg 接口操作参数

输入参数：

参 数	说 明
pushconsumer	stirng，PushConsumer的惟一标识，句柄，用来接收通知

输出/返回：

参 数	说 明
result	unsigned long，订购号

表 13 disNotReg 接口操作参数

输入参数：

参 数	说 明
nmsconsumerid	unsigned long，订购号

输出/返回：无

通知的组成有通知头和通知体，通知头含有如下3部分信息。通知体由数组构成，数组中的每个元素都是一个名值对，名字表示属性的含义，值表示属性的取值。不同的通知类型具有不同的通知体，通知体的组成在告警管理一节中描述。

表 14 通知头组成

属 性	名 称	描 述	类 型	限定符
Domain	域名		string	M,R/W
NoficationType	类型名		string	M,R/W
Eventname	事件名		string	M,R/W

8.2 配置管理

配置管理中主要提供了网元系统信息、端口配置（线路类型和线路编码）、禁用/启用端口、复位端口以及创建线路参数模板和告警参数模板等操作。

8.2.1 接口操作表

表 15 配置管理接口操作表

接口名	描 述
getNeConf	获取指定网元的相关属性
setNeConf	设置指定网元的相关属性
getAdslLineConf	获取指定ADSL线路配置信息

表15（续）

接口名	描　述
setAdslLineConf	设置指定ADSL线路配置信息
getAdslLineConfProfile	获取ADSL线路参数模板配置
crAdslLineConfProfile	创建ADSL线路参数模板
mdAdslLineConfProfile	修改ADSL线路参数模板
delAdslLineConfProfile	删除ADSL线路参数模板
getAdslAlarmProfile	获取指定ADSL告警参数模板配置
crAdslLineAlarmConfProfile	创建ADSL告警参数模板
mdAdslLineAlarmConfProfile	修改ADSL告警参数模板.
delAdslLineAlarmConfProfile	删除ADSL告警参数模板
getAdslLine2PlusConf	获取指定ADSL2+线路配置信息
setAdslLine2PlusConf	设置指定ADSL2+线路配置信息
getAdsl2PlusLineConfProfile	获取ADSL2+线路参数模板配置
crAdsl2PlusLineConfProfile	创建ADSL2+线路参数模板
mdAdsl2PlusLineConfProfile	修改ADSL2+线路参数模板
delAdsl2PlusLineConfProfile	删除ADSL2+线路参数模板
getAdsl2PlusAlarmProfile	获取指定ADSL2+告警参数模板配置
crAdsl2PlusLineAlarmConfProfile	创建ADSL2+告警参数模板
mdAdsl2PlusLineAlarmConfProfile	修改ADSL2+告警参数模板
delAdsl2PlusLineAlarmConfProfile	删除ADSL2+告警参数模板
getVdslLineConf	获取指定VDSL线路配置信息
setVdslLineConf	设置指定VDSL线路配置信息
getVdslLineConfProfile	获取VDSL线路参数模板配置
crVdslLineConfProfile	创建VDSL线路参数模板
mdVdslLineConfProfile	修改VDSL线路参数模板
delVdslLineConfProfile	删除VDSL线路参数模板
getVdslAlarmProfile	获取指定VDSL告警参数模板配置
crVdslLineAlarmConfProfile	创建VDSL告警参数模板
mdVdslLineAlarmConfProfile	修改VDSL告警参数模板
delVdslLineAlarmConfProfile	删除VDSL告警参数模板
getShdslLineConf	获取指定SHDSL线路配置信息
setShdslLineConf	设置指定SHDSL线路配置信息
getHdsl2ShdslEndpoint	获取指定SHDSL端点配置（告警模板名）
setHdsl2ShdslEndpoint	设置指定SHDSL端点配置
getShdslLineConfProfile	获取SHDSL线路参数模板配置

表15（续）

接口名	描　述
crShdslLineConfProfile	创建SHDSL线路参数模板
mdShdslLineConfProfile	修改SHDSL线路参数模板
delShdslLineConfProfile	删除SHDSL线路参数模板
getShdslAlarmProfile	获取指定SHDSL告警参数模板配置
crShdslLineAlarmConfProfile	创建SHDSL告警参数模板
mdShdslLineAlarmConfProfile	修改SHDSL告警参数模板
delShdslLineAlarmConfProfile	删除ASHDSL告警参数模板
disablePort	禁用指定端口
enablePort	启用指定端口
resetPort	复位指定端口

8.2.2 接口操作参数和结果说明

表 16　getNeConf 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识

输出/返回：

参　数	说　明
result	NeConf，网元相关属性

异常：

参　数	说　明
exception	NeNotExistException

表 17　NeConf 组成

属　性	名　称	描　述	类　型	限定符
readattr	网元只读属性		ReadOnlyNeConf	M,R
setattr	网元可写属性		CanSetNeConf	M,R/W

表 18　ReadOnlyNeConf 组成

属　性	名　称	描　述	类　型	限定符
sysDescr	系统描述		String	M,R
sysObjectID	系统 ObjectId		String	M,R
sysUpTime	系统启动时间		Long	M,R

表 19　CanSetNeConf 组成

属　性	名　称	描　述	类　型	限定符
sysName	系统名称		String	M,R/W
sysLocation	系统位置		String	M,R/W
sysContact	系统联系方式		String	M,R/W

表 20　setNeConf 接口操作参数

输入参数：

参　数	说　明
neid	String，网元标识
conf	CanSetNeConf，可配置的网元属性

输出/返回：

参　数	说　明
result	boolean，表示配置是否成功

异常：

参　数	说　明
exception	NeNotExistException

表 21　getAdslLineConf 接口操作参数

输入参数：

参　数	说　明
neid	string,网元Id
cardid	short，指定卡标识
portid	short，指定端口标识

输出/返回：

参　数	说　明
result	AdslLineConf，端口相关配置信息

异常：

参　数	说　明
exception	NeNotExistException IlleagalParaException

表 22　AdslLineConf 组成

属　性	名　称	描　述	类　型	限定符
adslLineCoding	线路编码		enum LineCoding{ other, dmt, cap, qam, glite　　};	M,R/W
adslLineType	线路类型		enum LineType{ noChannel, fastOnly, interleavedOnly, fastOrInterleaved, fastAndInterleaved　　};	M,R/W

表 22（续）

属 性	名 称	描 述	类 型	限定符
adslLineConfProfile	线路配置参数模板名		string	M,R/W
adslLineAlarmConfProfile	线路告警参数模板名		string	M,R/W
username	线路所属用户标识		string	M,R/W
userphone	线路所属用户		string	O,R/W
ifOperStatus	线路的运行状态		enum IfStatusType { up, down, testing };	M,R
ifAdminStatus	线路的管理状态		enum IfStatusType { up, down, testing };	M,R/W

表 23 setAdslLineConf 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
cardno	short，指定卡标识
portno	short，指定端口标识
conf	AdslLineConf，设置其中的可写属性

输出/返回：

参 数	说 明
result	boolean,线路设置是否成功

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 24 getAdslLineConfProfile 接口操作参数

输入参数:

参 数	说 明
neid	string，网元标识
lineprofilename	Adsl 线路参数模板名

输出/返回：

参 数	说 明
result	AdslLineConfProfileTable

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 25 AdslLineConfProfileTable 组成

属 性	名 称	描 述	类 型	限定符
adslLineConfProfileName	线路参数配置模板名称	32 byte	string	M,R/W
adslAtucConfRateMode	Atuc 速率模式		enum RateMode { fixedmode, adaptAtStartup, adaptAtRuntime };	M,R/W
adslAtucConfRateChanRatio	Atuc 速率比	取值范围：0~100	short	M,R/W
adslAtucConfTargetSnrMgn	Atuc 目标信噪比	取值范围：0~310， 单位：tenth dB	short	M,R/W
adslAtucConfMaxSnrMgn	Atuc 最大信噪比	取值范围：0~310， 单位：tenth dB	short	M,R/W
adslAtucConfMinSnrMgn	Atuc 最小信噪比	取值范围：0~310， 单位：tenth dB	short	M,R/W
adslAtucConfDownshiftSnrMgn	Atuc 信噪比裕度减速限	取值范围：0~310， 单位：tenth dB	short	M,R/W
adslAtucConfUpshiftSnrMgn	Atuc 信噪比裕度提速限	取值范围：0~310， 单位 tenth dB	short	M,R/W
adslAtucConfMinUpshiftTime	Atuc 信噪比裕度最小速率上调时间	取值范围：0~16383， 单位：seconds	short	M,R/W
adslAtucConfMinDownshiftTime	Atuc 信噪比裕度最小速率下调时间	取值范围：0~16383， 单位：seconds	short	M,R/W
adslAtucChanConfFastMinTxRate	Atuc 快速信道最小速率	单位：bit/s	long	M,R/W
adslAtucChanConfInterleaveMinTxRate	Atuc 交织信道最小速率	单位：bit/s	long	M,R/W
adslAtucChanConfFastMaxTxRate	Atuc 快速信道最大速率	单位：bit/s	long	M,R/W
adslAtucChanConfInterleaveMaxTxRate	Atuc 交织信道最大速率	单位：bit/s	long	M,R/W
adslAtucChanConfMaxInterleaveDelay	Atuc 最大交织延迟	范围：0~255， 单位：milli-seconds	short	M,R/W

表 25（续）

属 性	名 称	描 述	类 型	限定符
adslAturConfRateMode	Atur 速率模式		enum RateMode{ fixedmode, adaptAtStartup, adaptAtRuntime };	M,R/W
adslAturConfRateChanRatio	Atur 速率比	取值范围：0~100	short	M,R/W
adslAturConfTargetSnrMgn	Atur 目标信噪比	取值范围：0~310，单位：tenth dB	short	M,R/W
adslAturConfMaxSnrMgn	Atur 最大信噪比	取值范围：0~310，单位：tenth dB	short	M,R/W
adslAturConfMinSnrMgn	Atur 最小信噪比	取值范围：0~310，单位：tenth dB	short	M,R/W
adslAturConfDownshiftSnrMgn	Atur 信噪比裕度减速限	取值范围：0~310，单位：tenth dB	short	M,R/W
adslAturConfUpshiftSnrMgn	Atur 信噪比裕度提速限	取值范围 0~310，单位 tenth dB	short	M,R/W
adslAturConfMinUpshiftTime	Atur 信噪比裕度最小速率上调时间	取值范围：0~16383，单位：seconds	short	M,R/W
adslAturConfMinDownshiftTime	Atur 信噪比裕度最小速率下调时间	取值范围：0~16383，单位：seconds	short	M,R/W
AdslAturChanConfFastMinTxRate	Atur 快速信道最小速率	单位：bit/s	long	M,R/W
AdslAturChanConfInterleaveMinTxRate	Atur 交织信道最小速率	单位：bit/s	long	M,R/W
AdslAturChanConfFastMaxTxRate	Atur 快速信道最大速率	单位：bit/s	long	M,R/W
adslAturChanConfInterleaveMaxTxRate	Atur 交织信道最大速率	单位：bit/s	long	M,R/W
adslAturChanConfMaxInterleaveDelay	Atur 最大交织延迟	范围：0~255，单位：milli-seconds	short	M,R/W

表 26 crAdslLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	String，网元标识
linconf	AdslLineConfProfileTable,线路参数模板

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 27 mdAdslLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	String，网元标识
linconf	AdslLineConfProfileTable,线路参数模板

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 28 delAdslLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	String，网元标识
adslLineConfProfileName	string，线路参数模板名称

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException

表 29 getAdslAlarmProfile 接口操作参数表

输入参数：

参 数	说 明
neid	string，网元标识
alarmprofilename	string，告警参数模板名称

输出/返回：

参 数	说 明
result	AdslLineAlarmConfProfileTable

异常：

参 数	说 明
exception	NeNotExistException IlleagalParaException

表 30 AdslLineAlarmConfProfileTable 组成

属 性	名 称	描 述	类 型	限定符
adslLineAlarmConfProfileName	告警参数模板名	32 bytes	string	M,R/W
adslAtucThresh15MinLofs	Atuc15min 帧丢失阈值	范围：0~900， 单位：seconds	short	M,R/W
adslAtucThresh15MinLoss	Atuc15min 信号丢失阈值	范围：0~900， 单位：seconds	short	M,R/W
adslAtucThresh15MinLols	Atuc15min 链路丢失阈值	范围：0~900， 单位：seconds	short	M,R/W
adslAtucThresh15MinLprs	Atuc15min 电源阈值	范围：0~900， 单位：seconds	short	M,R/W
adslAtucThresh15MinESs	Atuc15min 错误秒阈值	范围：0~900， 单位：seconds	short	M,R/W
adslAtucThreshFastRateUp	Atuc 快速信道速率递增量告警阈值	单位：bit/s	long	M,R/W
adslAtucThreshInterleaveRateUp	Atuc 交织信道速率递增量告警阈值	单位：bit/s	long	M,R/W
adslAtucThreshFastRateDown	Atuc 快速信道速率递减量告警阈值	单位：bit/s	long	M,R/W
adslAtucThreshInterleaveRateDown	Atuc 交织信道速率递减量告警阈值	单位：bit/s	long	M,R/W
adslAtucInitFailureTrapEnable	初始化失败告警	true:enable(1); false: disable(2)	short	M,R/W
adslAturThresh15MinLofs	Atur15min 帧丢失阈值	范围：0~900， 单位：seconds	short	M,R/W
adslAturThresh15MinLoss	Atur15min 信号丢失阈值	范围：0~900， 单位：seconds	short	M,R/W
adslAturThresh15MinLprs	Atur15min 电源阈值	范围：0~900， 单位：seconds	short	M,R/W
adslAturThresh15MinESs	Atur15min 错误秒阈值	范围：0~900， 单位：seconds	short	M,R/W
adslAturThreshFastRateUp	Atur 快速信道速率递增量告警阈值	单位：bit/s	long	M,R/W
adslAturThreshInterleaveRateUp	Atur 交织信道速率递增量告警阈值	单位：bit/s	long	M,R/W
adslAturThreshFastRateDown	Atur 快速信道速率递减量告警阈值	单位：bit/s	long	M,R/W
adslAturThreshInterleaveRateDown	Atur 交织信道速率递减量告警阈值	单位：bit/s	long	M,R/W

表 31　crAdslLineAlarmConfProfile 接口操作参数

输入参数：

参　数	说　明
neid	string,网元Id
alarmonf	AdslLineAlarmConfProfileTable，告警参数模板

输出/返回：

参　数	说　明
result	boolean

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 32　mdAdslLineAlarmConfProfile 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
alarmonf	AdslLineAlarmConfProfileTable，告警参数模板

输出/返回：

参　数	说　明
result	boolean

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 33　delAdslLineAlarmConfProfile 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
alarmprofilename	string，告警参数模板名称

输出/返回：

参　数	说　明
result	boolean

异常：

参　数	说　明
exception	NeNotExistException

表 34 getAdsl2PlusLineConf 接口操作参数

输入参数：

参 数	说 明
neid	String，网元 Id
cardid	short，指定卡标识
portid	short，指定端口标识

输出/返回：

参 数	说 明
result	Adsl2PlusLineConf，端口相关配置信息

异常：

参 数	说 明
exception	NeNotExistException IlleagalParaException

表 35 Adsl2PlusLineConf 组成

属 性	名 称	描 述	类 型	限定符
adslLineConfProfile	线路配置参数模板名		String	M,R/W
adslLineAlarmConfProfile	线路告警参数模板名		String	M,R/W
username	线路所属用户标识		String	M,R/W
userphone	线路所属用户		String	O,R/W
ifOperStatus	线路的运行状态		enum IfStatusType { up, down, testing };	M,R
ifAdminStatus	线路的管理状态		enum IfStatusType { up, down, testing };	M,R/W
adslLineTransAtucConfig	配置传输模式	enumTransMode{ansit1413, etsi, q9921PotsNonOverlapped, q9921PotsOverlapped, q9921IsdnNonOverlapped, q9921isdnOverlapped, q9921tcmIsdnNonOverlapped, q9921tcmIsdnOverlapped, q9922potsNonOverlapeed, q9922potsOverlapped, q9922tcmIsdnNonOverlapped, q9922tcmIsdnOverlapped, q9921tcmIsdnSymmetric };	Sequence of TransMode	M,R/W
adslLineTransAtucCap	该线路支持的工作模式	同上	Sequence of TransMode	M,R
adslLineTransAtucActual	该线路实际的工作模式	同上	Sequence of TransMode	M,R

表 36 setAdsl2PlusLineConf 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
cardid	short，指定卡标识
portid	short，指定端口标识
lineconf	Adsl2PlusLineConf，可以设置的配置信息，其中的只读属性不设置

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 37 getAdsl2PlusLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
lineprofilename	string,adsl2plus线路参数模板名

输出/返回：

参 数	说 明
result	Adsl2PlusLineConfProfileTable，参数模板信息

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 38 Adsl2PlusLineConfProfileTable 组成

属 性	名 称	描 述	类 型	限定符
adsllineprf	Adslline 线路配置		AdslLineConfProfileTable	M,R/W

adslConfProfileLineType	线路类型		enum LineType{
			noChannel,
			fastOnly,
			interleavedOnly,
			fastOrInterleaved,
			fastAndInterleaved
			};
				M,R/W

表 39 crAdsl2PlusLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
linconf	Adsl2PlusLineConfProfileTable, adsl2plus线路参数模板

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 40 mdAdsl2PlusLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
linconf	Adsl2PlusLineConfProfileTable, adsl2plus线路参数模板

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 41 delAdsl2PlusLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
lineprofilename	string, adsl2plus线路参数模板

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException

表 42 getAdsl2PlusAlarmProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
alarmprofilename	string,adsl2plus告警参数模板名

输出/返回：

参 数	说 明
result	Adsl2PlusLineAlarmConfProfileTable，告警参数模板

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

43 Adsl2PlusLineConfProfileTable 组成表

属 性	名 称	描 述	类 型	限定符
adslalarmprf	Adsl 告警配置模板		AdslLineConf ProfileTable	M,R/W
adslAtucThreshold15MinFailedFastR	Atuc 快速训练失败告警门限	取值范围：0~900， 单位：seconds	Short	M,R/W
adslAtucThreshold15MinSesL	Atuc 线路严重误码告警门限	取值范围：0~900， 单位：seconds	Short	M,R/W
adslAtucThreshold15MinUasL	Atuc 线路不可用告警门限	取值范围：0~900， 单位：seconds	Short	M,R/W
adslAturThreshold15MinSesL	Atur 线路严重误码告警门限	取值范围：0~900， 单位：seconds	Short	M,R/W
adslAturThreshold15MinUasL	Atur 线路不可用告警门限	取值范围：0~900， 单位：seconds	Short	M,R/W

表 44 crAdsl2PlusLineAlarmConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
alarmconf	Adsl2PlusLineAlarmConfProfileTable, adsl2plus告警参数模板

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 45　mdAdsl2PlusLineAlarmConfProfile 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
alarmconf	Adsl2PlusLineAlarmConfProfileTable, adsl2plus告警参数模板

输出/返回：

参　数	说　明
result	boolean

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 46　delAdsl2PlusLineAlarmConfProfile 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
alarmprofilename	string, adsl2plus告警参数模板名

输出/返回：

参　数	说　明
result	boolean

异常：

参　数	说　明
exception	NeNotExistException

表 47　getVdslLineConf 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
cardid	short，板卡标识
portid	short，端口标识

输出/返回：

参　数	说　明
result	VdslLineConf，Vdsl线路配置

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 48 VdslLineConf 组成

属 性	名 称	描 述	类 型	限定符
vdslLineType	Vdsl 线路类型		enum LineType{ noChannel, fastOnly, interleavedOnly, fastOrInterleaved, fastAndInterleave};	M,R/W
lineCoding	Vdsl 线路编码		enum VdslLineCoding{ othercoding, mcm, scm };	M,R/W
ifAdminStatus	线路管理状态		enum IfStatusType { up, down, testing };	M,R/W
ifOperStatus	线路运行状态		enum IfStatusType { up, down, testing };	M,R
vdslLineConfProfile	Vdsl 线路参数模板名		string	M,R/W
vdslLineAlarmConfProfile	Vdsl 告警参数模板名		string	M,R/W
username	线路所属用户		string	M,R/W
userphone	用户电话		string	O,R/W

表 49 setVdslLineConf 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
cardid	short，板卡标识
portid	short，端口标识
lineconf	vdslLineConf，Vdsl线路配置

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 50 getVdslLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
lineprofilename	线路参数模板名

输出/返回：

参 数	说 明
result	VdslLineConfProfileTable，Vdsl线路参数模板集

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 51 VdslLineConfProfileTable 组成

属 性	名 称	描 述	类 型	限定符
vdslLineConfProfileName	线路参数模板名		string	M,R/W
vdslLineConfDownRateMode	下行速率模式		enum VdslRateMode{ manualRateMode, adaptAtInit };	M,R/W
vdslLineConfUpRateMode	上行速率模式		同上	M,R/W
vdslLineConfDownMaxPwr	下行最大功率	范围：0~58， 单位：0.25dBm	short	M,R/W
vdslLineConfUpMaxPwr	上行最大功率	范围：0~58， 单位：0.25dBm	short	M,R/W
vdslLineConfDownMaxSnrMgn	下行最大信噪比	范围：0~127， 单位：0.25dB	short	M,R/W
vdslLineConfDownMinSnrMgn	下行最小信噪比	范围：0~127， 单位：0.25dB	short	M,R/W
vdslLineConfDownTargetSnrMgn	下行目标信噪比	范围：0~127， 单位：0.25dB	short	M,R/W
vdslLineConfUpMaxSnrMgn	上行最大信噪比	范围：0~127， 单位：0.25dB	short	M,R/W
vdslLineConfUpMinSnrMgn	上行最小信噪比	范围：0~127， 单位：0.25dB"	short	M,R/W
vdslLineConfUpTargetSnrMgn	上行目标信噪比	范围：0~127， 单位：0.25dB	short	M,R/W
vdslLineConfDownFastMaxDataRate	下行快速信道最大速率		long	M,R/W

表 51（续）

属　性	名　称	描　述	类　型	限定符
vdslLineConfDownFastMinDataRate	下行快速信道最小速率		long	M,R/W
vdslLineConfDownSlowMaxDataRate	下行慢速信道最大速率		long	M,R/W
vdslLineConfDownSlowMinDataRate	下行慢速信道最小速率		long	M,R/W
vdslLineConfUpFastMaxDataRate	上行快速信道最大速率		long	M,R/W
vdslLineConfUpFastMinDataRate	上行快速信道最小速率		long	M,R/W
vdslLineConfUpSlowMaxDataRate	上行慢速信道最大速率		long	M,R/W
vdslLineConfUpSlowMinDataRate	上行慢速信道最小速率		long	M,R/W
vdslLineConfDownRateRatio	下行速率比率	范围：0~100， 单位：%	short	M,R/W
vdslLineConfUpRateRatio	上行速率比率	范围：0~100， 单位：%	short	M,R/W
vdslLineConfDownMaxInterDelay	下行最大交织延迟	范围：0~255， 单位：milli-seconds	short	M,R/W
vdslLineConfUpMaxInterDelay	上行最大交织延迟	范围：0~255 单位：milli-seconds	short	M,R/W
vdslLineConfDownPboControl	下行功率补偿控制		enum VdslPboControl{ disabled, auto, manualPboContro}	M,R/W
vdslLineConfUpPboControl	上行功率补偿控制		enum VdslPboControl{ disabled, auto, manualPboContro}	M,R/W
vdslLineConfDownPboLevel	下行功率补偿等级	范围：0~160		M,R/W
vdslLineConfUpPboLevel	上行功率补偿等级	范围：0~160		M,R/W

表 52　crVdslLineConfProfile 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
lineconf	VdslLineConfProfileTable，线路参数模板

输出/返回：

参　数	说　明
result	boolean

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 53　mdVdslLineConfProfile 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
lineconf	VdslLineConfProfileTable，线路参数模板

输出/返回：

参　数	说　明
result	boolean

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 54　delVdslLineConfProfile 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
lineprofilename	string，线路参数模板名

输出/返回：

参　数	说　明
result	boolean

异常：

参　数	说　明
exception	NeNotExistException

表 55　getVdslAlarmProfile 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
alarmprofilename	string，告警参数模板名

输出/返回：

参　数	说　明
result	VdslLineAlarmConfProfileTable

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 56　VdslLineAlarmConfProfileTable 组成

属　性	名　称	描　述	类　型	限定符
vdslLineAlarmConfProfileName	告警参数模板名	32bytes	string	M,R/W
vdslLineAlarmConfThresh15MinLofs	15min 帧丢失最小阈值	范围：0~900	short	M,R/W
vdslLineAlarmConfThresh15MinLoss	15min 信号丢失最小阈值	范围：0~900	short	M,R/W
vdslLineAlarmConfThresh15MinLprs	15min 电源丢失最小阈值	范围：0~900	short	M,R/W
vdslLineAlarmConfThresh15MinLols	15min 链路丢失最小阈值	范围：0~900	short	M,R/W
vdslLineAlarmConfThresh15MinESs	15min 错误秒最小阈值	范围：0~900	short	M,R/W
vdslLineAlarmConfThresh15MinUASs	15min 不可达秒最小阈值	范围：0~900	short	M,R/W
vdslLineAlarmConfInitFailure	初始化失败阈值	范围：0~900	short	M,R/W

表 57　crVdslLineAlarmConfProfile 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
alarmconf	VdslLineAlarmConfProfileTable，告警参数模板

输出/返回：

参　数	说　明
result	boolean

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 58　mdVdslLineConfProfile 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
alarmconf	VdslLineAlarmConfProfileTable，告警参数模板

输出/返回：

参　数	说　明
result	boolean

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 59 delVdslLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
alarmprofilename	string，告警参数模板名

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException

表 60 getShdslLineConf 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
cardid	short，板卡标识
portid	short，端口标识

输出/返回：

参 数	说 明
result	Hdsl2ShdslSpanConfTable，Shdsl线路配置

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 61 Hdsl2ShdslSpanConfTable 组成

属 性	名 称	描 述	类 型	限定符
hdsl2ShdslSpanConfNumRepeaters	可以配置的中继器数目	范围：0~8	short	M,R/W
hdsl2ShdslStatusNumAvailRepeaters	可支持的中继器数目	范围：0~8	short	M,R/W
ifAdminStatus	线路管理状态		enum IfStatusType { up, down, testing };	M,R/W
ifOperStatus	线路运行状态		enum IfStatusType { up, down, testing };	M,R

表 61（续）

属　性	名　称	描　述	类　型	限定符
vdslLineConfProfile	Vdsl 线路参数模板名		string	M,R/W
vdslLineAlarmConfProfile	Vdsl 告警参数模板名		string	M,R/W
username	线路所属用户		string	M,R/W
userphone	用户电话		string	O,R/W

表 62　setShdslLineConf 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
cardid	short，板卡标识
portid	short，端口标识
lineconf	Hdsl2ShdslSpanConfTable，Shdsl线路配置

输出/返回：

参　数	说　明
result	boolean

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 63　getHdsl2ShdslEndpoint 接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
pointconf	inout Hdsl2ShdslSpanConfTable，Shdsl线路配置，用Hdsl2ShdslSpanConfTable中的cardid，portid，hdsl2ShdslInvIndex，hdsl2ShdslEndpointSide，hdsl2ShdslEndpointWirePair一起作为索引取出hdsl2ShdslEndpointAlarmConfProfile

输出/返回：

参　数	说　明
endalarmprf	inout Hdsl2ShdslSpanConfTable，Shdsl线路配置，用Hdsl2ShdslSpanConfTable中的cardid，portid，hdsl2ShdslInvIndex，hdsl2ShdslEndpointSide，hdsl2ShdslEndpointWirePair一起作为索引取出hdsl2ShdslEndpointAlarmConfProfile

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 64 Hdsl2ShdslSpanConfTable 组成

属 性	名 称	描 述	类 型	限定符
cardid	板卡标识		short	M,R
portid	端口标识		short	M,R
hdsl2ShdslInvIndex	端点号	xtuC = 1,xtuR = 2,xru1 = 3, xru2 = 4,xru3 = 5,xru4 = 6, xru5 = 7,xru6 = 8,xru7 = 9, xru8 = 10，其中 3~10 根据可支持的中继器数目来定	short	M,R
hdsl2ShdslEndpointSide	端点侧	networkSide =1,customerSide =2， 对 xtuc 来说，只有 customerSide， 对 xtur 来说，只有 networkSide	short	M,R
hdsl2ShdslEndpointWirePair	端点线对	wirePair1 = 1 wirePair2 = 2	short	M,R
hdsl2ShdslEndpointAlarmConfProfile	端点告警参数模板名		string	M,R/W

表 65 setHdsl2ShdslEndpoint 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
conf	Hdsl2ShdslEndpointConfTable，端点配置参数

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 66 getShdslLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
lineprofilename	线路参数模板名

输出/返回：

参 数	说 明
result	ShdslLineConfProfileTable，Shdsl线路参数模板集

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表 67　ShdslLineConfProfileTable 组成

属　性	名　称	描　述	类　型	限定符
hdsl2ShdslSpanConfProfileName	线路参数模板名		string	M,R/W
hdsl2ShdslSpanConfWireInterface	线制	TwoWire = 1 fourWire = 2	short	M,R/W
hdsl2ShdslSpanConfMinLineRate	最小速率		long	M,R/W
hdsl2ShdslSpanConfMaxLineRate	最大速率		long	M,R/W
hdsl2ShdslSpanConfPSD	功率谱密度	symmetric = 1 asymmetric = 2	short	M,R/W
hdsl2ShdslSpanConfTransmission Mode	传输模式	enum ShdslSpanConf TransmissionMode {Region1, Region2}	sequence of ShdslSpanConfTrans missionModes	M,R/W
hdsl2ShdslSpanConfRemoteEnabled	远程管理使能	enable = 1 disable = 2	short	M,R/W
hdsl2ShdslSpanConfPowerFeeding	电源馈电	noPower = 1, powerFeed = 2, wettingCurrent = 3	short	M,R/W
hdsl2ShdslSpanConfCurrCondTarget MarginDown	当前下行目标信噪比裕度	范围：－10~21， 单位：dB	int	M,R/W
hdsl2ShdslSpanConfWorstCaseTarget MarginDown	最糟情况下下行目标信噪比裕度	范围：－10~21， 单位：dB	int	M,R/W
hdsl2ShdslSpanConfCurrCondTarget MarginUp	当前上行目标信噪比裕度	范围：－10~21， 单位：dB	int	M,R/W
hdsl2ShdslSpanConfWorstCaseTarget MarginUp	最糟情况下上行目标信噪比裕度	范围：－10~21， 单位：dB	int	M,R/W

hdsl2ShdslSpanConfUsedTarget Margins	应用的目标信噪比裕度	enum ShdslSpanConf UsedTargetMargin {currCondDown,
		//第 0 位 worstCaseDown,
		//第 1 位 currCondUp,
		//第 2 位 worstCaseUp
		//第 3 位};
			sequence of Shdsl SpanConfUsedTarget Margin	M,R/W

hdsl2ShdslSpanConfReferenceClock	参考时钟	localClk = 1, networkClk = 2, dataOrNetworkClk = 3, dataClk = 4	short	M,R/W
hdsl2ShdslSpanConfLineProbeEnable	线路探测使能	enable = 1, disable = 2	short	M,R/W

表 68 crShdslLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
lineconf	ShdslLineConfProfileTable，线路参数模板

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 69 mdShdslLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
lineconf	ShdslLineConfProfileTable，线路参数模板

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 70 mdShdslLineConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
lineprofilename	string，线路参数模板名

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException,

表 71 getShdslAlarmProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
lineprofilename	告警参数模板名

输出/返回：

参 数	说 明
result	ShdslLineAlarmConfProfileTable，Shdsl告警参数模板集

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 72 ShdslLineAlarmConfProfileTable

属 性	名 称	描 述	类 型	限定符
hdsl2ShdslEndpointAlarmConfProfileName	告警参数模板名	32byte	string	M,R/W
hdsl2ShdslEndpointThreshLoopAttenuation	环路衰减阈值	范围：+127~128， 单位：0.1dB	int	M,R/W
hdsl2ShdslEndpointThreshSNRMargin	信噪比裕度阈值	范围：+127~128， 单位：0.1dB	int	M,R/W
hdsl2ShdslEndpointThreshES	误码秒阈值	范围：0~900， 单位：seconds	short	M,R/W
hdsl2ShdslEndpointThreshSES	严重误码秒阈值	范围：0~900， 单位：seconds	short	M,R/W
hdsl2ShdslEndpointThreshCRCanomalies	CRC 检验异常数阈值	范围：0~900， 单位：seconds	short	M,R/W
hdsl2ShdslEndpointThreshLOSWS	同步字丢失秒数阈值	范围：0~900， 单位：seconds	short	M,R/W
hdsl2ShdslEndpointThreshUAS	不可达秒阈值	范围：0~900， 单位：seconds	short	M,R/W

表 73 crShdslLineAlarmConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
alarmconf	ShdslLineAlarmConfProfileTable，告警参数模板

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 74 mdShdslLineAlarmConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
alarmconf	ShdslLineAlarmConfProfileTable，告警参数模板

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 75 delShdslLineAlarmConfProfile 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
alarmprofilename	string，告警参数模板名

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	NeNotExistException,

表 76 disablePort 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
cardid	short，指定卡标识
portid	short，指定端口标识

输出/返回：

参 数	说 明
result	boolean，操作是否成功

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 77 enablePort 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
cardid	short，指定卡标识
portid	short，指定端口标识

输出/返回：

参 数	说 明
result	boolean,操作是否成功

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 78 resetPort 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
cardid	short，指定卡标识
portid	short，指定端口标识

输出/返回：

参 数	说 明
result	boolean，操作是否成功

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

8.3 告警管理

告警管理提供了 EMS 产生的和网元产生的告警的自动上报功能，告警的同步功能，以及告警过滤器的设置来防止消息洪泛。告警上报采用通知的组织形式。

8.3.1 接口操作表

表 79 告警管理接口操作

接口名	描 述
acknowledge_alarms	确认指定告警
unacknowledge_alarms	取消确认指定告警
comment_alarms	说明告警
clear_alarms	清除告警
get_alarm_list	获取未被清除的所有告警信息
get_alarm_count	获取未被清除的不同级别告警数目

8.3.2 接口操作参数和结果说明

表 80 NotifyNewAlarm 组成

通知头	
属 性	取值
Domain	Alarm
NoficationType	NotificationType.NOTIFY_FM_NEW_ALARM

EventName	Enum AlarmType{
	COMMUNICATIONS_ALARM,
	PROCESSING_ERROR_ALARM
	ENVIRONMENTAL_ALARM
	QUALITY_OF_SERVICE_ALARM
	EQUIPMENT_ALARM
	}

通知体			
属 性	取 值	描 述	是否可过滤条件
NeID	string	告警源，告警产生的资源ID，为网元ID	是
Netype	string	告警源类型，为网元类型	是
PERCEIVED_SEVERITY	Enum PerceivedSeverity{ INDETERMINATE; CRITICAL; MAJOR; MINOR; WARNING; CLEARED;};	告警级别	是

表 81（续）

属　性	取　值	描　述	是否可过滤条件
PROBABLE_CAUSE	Enum DslProbCause{ColdStart, AuthenticationFailure,LinkDown, LinkUp,AdslAtucTraps,AdslAturTraps, CardUp ,CardDown,FanOpenStatusChange, PowerStatusChange,NeAdded,NeDeleted, SplitterCardUp,SplitterCardDown}	告警原因	是
EVENT_TIME	OMG 的 UTC 类型	告警时间	是
ENTITY_ID	string	实体标识，用来可以惟一标识一个告警的实体，以便清除告警时确认清除的是何条告警	否
ALARM_ID	string	告警的惟一标识	否

表 81　notifyComments 组成

通知头	
属　性	取　值
Domain	Alarm
NoficationType	NotificationType.NOTIFY_FM_COMMENT_ADDED

EventName	Enum AlarmType{
	COMMUNICATIONS_ALARM,
	PROCESSING_ERROR_ALARM
	ENVIRONMENTAL_ALARM
	QUALITY_OF_SERVICE_ALARM
	EQUIPMENT_ALARM
	}

通知体			
属　性	取　值	描　述	是否可过滤条件
NeID	string	告警源，告警产生的资源ID，为网元 ID	是
Netype	string	告警源类型，为网元类型	是
PERCEIVED_SEVERITY	Enum PerceivedSeverity{ INDETERMINATE; CRITICAL; MAJOR; MINOR; WARNING; CLEARED;};	告警级别	是

表 81（续）

属　性	取　值	描　述	是否可过滤条件
PROBABLE_CAUSE	Enum DslProbCause{ColdStart, AuthenticationFailure,LinkDown, LinkUp,AdslAtucTraps,AdslAturTraps, CardUp ,CardDown,FanOpenStatusChange, PowerStatusChange,NeAdded,NeDeleted, SplitterCardUp,SplitterCardDown}	告警原因	是
EVENT_TIME	OMG 的 UTC 类型	告警时间	是
ENTITY_ID	string	实体标识，用来可以惟一标识一个告警的实体，以便清除告警时确认清除的是何条告警	否
ALARM_ID	string	告警的惟一标识	否
COMMENTS	string	告警的注释，说明	否

表 82　notifyAckStateChanged 组成

通知头	
属　性	取　值
Domain	Alarm
NoficationType	NotificationType. NOTIFY_FM_ACK_STATE_CHANGED

EventName	Enum AlarmType{
	COMMUNICATIONS_ALARM,
	PROCESSING_ERROR_ALARM
	ENVIRONMENTAL_ALARM
	QUALITY_OF_SERVICE_ALARM
	EQUIPMENT_ALARM
	}

通知体			
属　性	取　值	描　述	是否可过滤条件
NeID	string	告警源，告警产生的资源ID，为网元 ID	是
Netype	string	告警源类型，为网元类型	是
PERCEIVED_SEVERITY	Enum PerceivedSeverity{ INDETERMINATE; CRITICAL; MAJOR; MINOR; WARNING; CLEARED;};	告警级别	是

表 82（续）

属　性	取　值	描　述	是否可过滤条件
PROBABLE_CAUSE	Enum DslProbCause{ColdStart, AuthenticationFailure,LinkDown, LinkUp,AdslAtucTraps,AdslAturTraps, CardUp ,CardDown,FanOpenStatusChange, PowerStatusChange,NeAdded,NeDeleted, SplitterCardUp,SplitterCardDown}	告警原因	是
EVENT_TIME	OMG 的 UTC 类型	告警时间	是
ENTITY_ID	string	实体标识，用来可以惟一标识一个告警的实体，以便清除告警时确认清除的是何条告警	否
ALARM_ID	string	告警的惟一标识	否
ACK_USER_ID	string	改变确认状态的用户标识	否
ACK_SYSTEM_ID	string	改变确认状态的系统标识	否
ACK_STATE	Enum AckState{ ACKNOWLEDGED, UNACKNOWLEDGED }	告警的当前确认状态	否

表 83　notifyClearedAlarm 组成

通知头	
属　性	取　值
Domain	Alarm
NoficationType	NotificationType. NOTIFY_FM_CLEARED_ALARM
EventName	Enum AlarmType{ COMMUNICATIONS_ALARM, PROCESSING_ERROR_ALARM ENVIRONMENTAL_ALARM QUALITY_OF_SERVICE_ALARM EQUIPMENT_ALARM }

通知体			
属　性	取　值	描　述	是否可过滤条件
NeID	string	告警源，告警产生的资源ID，为网元 ID	是
Netype	string	告警源类型，为网元类型	是

表 83（续）

属　性	取　值	描　述	是否可过滤条件
PERCEIVED_SEVERITY	Enum PerceivedSeverity{ INDETERMINATE; CRITICAL; MAJOR; MINOR; WARNING; CLEARED;};	告警级别	是
PROBABLE_CAUSE	Enum DslProbCause{ColdStart, AuthenticationFailure,LinkDown, LinkUp,AdslAtucTraps,AdslAturTraps, CardUp ,CardDown,FanOpenStatusChange, PowerStatusChange,NeAdded,NeDeleted, SplitterCardUp,SplitterCardDown}	告警原因	是
EVENT_TIME	OMG 的 UTC 类型	告警时间	是
ENTITY_ID	string	实体标识，用来可以惟一标识一个告警的实体，以便清除告警时确认清除的是何条告警	否
ALARM_ID	string	告警的惟一标识	否
CLEAR_USER_ID	string	清除告警的用户标识	否
CLEAR_SYSTEM_ID	string	清除告警的系统标识	否

表 84　acknowledge_alarms 接口操作参数

输入参数：

参　数	说　明
alarm_information_id_list	sequence of string，告警标识组成的列表
ack_user_id	string，确认用户标识
ack_system_id	string，确认系统标识（字符串为空，表明不指定）

输出/返回：

参　数	说　明
bad_ack_alarm_info_list	sequence of BadAcknowledgeAlarmInfo确认失败消息序列
result	enum Signal {OK, Failure, PartialFailure};

异常：

参　数	说　明
exception	AcknowledgeAlarms, ParameterNotSupported, IlleagalParaException

表 85 BadAcknowledgeAlarmInfo 组成

属 性	名 称	描 述	类 型	限定符
alarmid	告警标识	状态确认操作失败的告警标识	string	M,R
failure_category	失败类型		enum AcknowledgeFailureCategories {UNKNOWNALARMID, WRONGPERCEIVEDSEVERITY, ACKNOWLEDGMENTFAILED};	M,R
reason	失败原因		string	O,R

表 86 unacknowledge_alarms 接口操作参数

输入参数：

参 数	说 明
alarm_information_id_list	sequence of string，告警标识组成的列表
ack_user_id	string，确认用户标识
ack_system_id	string，确认系统标识（字符串为空，表明不指定）

输出/返回：

参 数	说 明
bad_alarm_information_id_list	sequence of BadAcknowledgeAlarmInfo，取消确认失败消息序列
result	enum Signal {OK, Failure, PartialFailure};

异常：

参 数	说 明
exception	UnacknowledgeAlarms, OperationNotSupported, ParameterNotSupported, IlleagalParaException

表 87 comment_alarms 接口操作参数

输入参数：

参 数	说 明
alarm_information_id_list	sequence of string，告警标识组成的列表
comment_user_id	string，说明用户标识
comment_system_id	string，说明系统标识（字符串为空，表明不指定）

输出/返回：

参 数	说 明
bad_alarm_information_id_list	sequence of BadAlarmInformationId，增加告警说明失败消息序列
result	enum Signal {OK, Failure, PartialFailure};

异常：

参　数	说　明
exception	CommentAlarms, OperationNotSupported, ParameterNotSupported, IlleagalParaException

表 88　BadAlarmInformationId 组成

属　性	名　称	描　述	类　型	限定符
alarmid	告警标识	状态确认操作失败的告警标识	string	M,R
reason	失败原因		string	M,R

表 89　clear_alarms 接口操作参数

输入参数：

参　数	说　明
alarm_information_id_list	sequence of string，告警标识组成的列表
clear_user_id	string，清除用户标识
clear_system_id	String，清除系统标识（字符串为空，表明不指定）

输出/返回：

参　数	说　明
bad_alarm_information_id_list	sequence of BadAlarmInformationId，清除告警失败消息序列
result	enum Signal {OK, Failure, PartialFailure};

异常：

参　数	说　明
exception	ClearAlarms, ParameterNotSupported, IlleagalParaException

表 90　get_alarm_list 接口操作参数

输入参数：

参　数	说　明
filter	string，过滤条件（字符串为空，表明不指定）

输出/返回：

参　数	说　明
result	sequence of QueryAlarmInfo，告警信息组成的列表，为了和告警上报的形式一致，采用和通知一样的组织来表示

异常：

参　数	说　明
exception	GetAlarmList, ParameterNotSupported, IlleagalParaException

表 91　QueryAlarmInfo 组成

通知头			
属　性	取　值		
Domain	Alarm		
NoficationType	get_alarm_list		

EventName	Enum AlarmType{
	COMMUNICATIONS_ALARM,
	PROCESSING_ERROR_ALARM
	ENVIRONMENTAL_ALARM
	QUALITY_OF_SERVICE_ALARM
	EQUIPMENT_ALARM
	}		

通知体			
属　性	取　值	描　述	是否可过滤条件
NeID	string	告警源，告警产生的资源 ID，为网元 ID	是
Netype	string	告警源类型，为网元类型	是
PERCEIVED_SEVERITY	Enum PerceivedSeverity{ INDETERMINATE; CRITICAL; MAJOR; MINOR; WARNING; CLEARED;};	告警级别	是
PROBABLE_CAUSE	Enum DslProbCause{ColdStart, AuthenticationFailure,LinkDown, LinkUp,AdslAtucTraps,AdslAturTraps, CardUp ,CardDown,FanOpenStatusChange, PowerStatusChange,NeAdded,NeDeleted, SplitterCardUp,SplitterCardDown}	告警原因	是
EVENT_TIME	OMG 的 UTC 类型	告警时间	是
ENTITY_ID	string	实体标识，用来可以惟一标识一个告警的实体，以便清除告警时确认清除的是何条告警	否
ALARM_CHANGED_TIME	OMG 的 UTC 类型	告警要是改变过，则含有这个属性	否

表 91（续）

属　性	取　值	描　述	是否可过滤条件
ALARM_ID	string	告警的惟一标识	否
ACK_USER_ID	string	改变确认状态的用户标识，告警若是曾经更改过确认状态，则含有这个属性	否
ACK_SYSTEM_ID	string	改变确认状态的系统标识，告警若是曾经更改过确认状态，则含有这个属性	否

ACK_STATE	Enum AckState{
	ACKNOWLEDGED, UNACKNOWLEDGED
	}
		告警当前的确认状态	否

表 92　get_alarm_count 接口操作参数

输入参数：

参　数	说　明
filter	String，过滤条件，为空表示不指定

输出/返回：

参　数	说　明
critical_count	out unsigned long，一级告警数目
major_count	out unsigned long，二级告警数目
minor_count	out unsigned long，三级告警数目
warning_count	out unsigned long，四级告警数目
indeterminate_count	out unsigned long，不确定级别告警数目

异常：

参　数	说　明
exception	GetAlarmCount, OperationNotSupported, ParameterNotSupported, IlleagalParaException

8.4　性能管理

性能管理中提供了对网元 C 端和 R 端实时、15min、当天和前一天的性能数据的采集。

8.4.1　接口操作表

表 93　性能管理接口操作

接口名	描　述
collectAdslCurrPerf	获取指定ADSL性能点的性能值
collectVdslCurrPerf	获取指定VDSL性能点的性能值
collectShdslCurrPerf	获取指定SHDSL性能点的性能值

8.4.2 接口操作参数和结果说明

表 94 collectAdslCurrPerf 接口操作参数

输入参数：

参 数	说 明
neid	string，网元标识
cardid	short，板卡标识
portid	short，端口标识
collectpoint	sequence of AdslPerfPoint，AdslPerfPoint 是 ADSL 性能点的枚举类型

输出/返回：

参 数	说 明
result	Sequence of long，性能值列表

异常：

参 数	说 明
exception	NeNotExistException, IlleagalParaException

表 95 AdslPerfPoint 枚举值

性能点	描 述
adslAtucCurrSnrMgn	ATUC当前噪声裕度
adslAtucCurrAtn	ATUC当前衰减
adslAtucCurrStatus	ATUC当前状态
adslAtucCurrOutputPwr	Atuc当前输出功率
adslAtucCurrAttainableRate	ATUC当前可达速率
adslAturCurrSnrMgn	ATUR当前噪声裕度
adslAturCurrAtn	ATUR衰减
adslAturCurrStatus	ATUR状态
adslAturCurrOutputPwr	ATUR输出功率
adslAturCurrAttainableRate	ATUR可达速率
adslAtucChanInterleaveDelay	交织延迟
adslAtucChanCurrTxRate	发送速率
adslAtucChanPrevTxRate	前次连接的发送速率
adslAturChanInterleaveDelay	交织延迟
adslAturChanCurrTxRate	发送速率
adslAturChanPrevTxRate	前次连接的发送速率
adslAtucPerfLofs	帧丢失数
adslAtucPerfLoss	信号丢失数
adslAtucPerfLols	链路丢失数

表 95（续）

性能点	描　述
adslAtucPerfLprs	电源丢失数
adslAtucPerfESs	误码秒数
adslAtucPerfInits	初始化失败次数
adslAtucPerfCurr15MinTimeElapsed	当前15min性能统计流逝的时间
adslAtucPerfCurr15MinLofs	当前15min内帧丢失数
adslAtucPerfCurr15MinLoss	当前15min内信号丢失数
adslAtucPerfCurr15MinLols	当前15min内链路丢失数
adslAtucPerfCurr15MinLprs	当前15min内电源丢失数
adslAtucPerfCurr15MinESs	当前15min内误码丢失数
adslAtucPerfCurr15MinInits	当前15min内初始化失败次数
adslAtucPerfCurr1DayTimeElapsed	当前1天性能统计流逝的时间
adslAtucPerfCurr1DayLofs	当前1天内帧丢失数
adslAtucPerfCurr1DayLoss	当前1天内信号丢失数
adslAtucPerfCurr1DayLols	当前1天内链路丢失数
adslAtucPerfCurr1DayLprs	当前1天内电源丢失数
adslAtucPerfCurr1DayESs	当前1天内误码丢失数
adslAtucPerfCurr1DayInits	当前1天内初始化失败次数
adslAtucPerfPrev1DayMoniSecs	前1天性能有效的统计时间
adslAtucPerfPrev1DayLofs	前1天帧丢失数
adslAtucPerfPrev1DayLoss	前1天信号丢失数
adslAtucPerfPrev1DayLols	前1天链路丢失数
adslAtucPerfPrev1DayLprs	前1天电源丢失数
adslAtucPerfPrev1DayESs	前1天误码丢失数
adslAtucPerfPrev1DayInits	前1天初始化失败次数
adslAturPerfLofs	帧丢失数
adslAturPerfLoss	信号丢失数
adslAturPerfLprs	电源丢失数
adslAturPerfESs	误码秒数
adslAturPerfCurr15MinTimeElapsed	当前15min性能统计流逝的时间
adslAturPerfCurr15MinLofs	当前15min内帧丢失数
adslAturPerfCurr15MinLoss	当前15min内信号丢失数
adslAturPerfCurr15MinLprs	当前15min内电源丢失数
adslAturPerfCurr15MinESs	当前15min内误码丢失数
adslAturPerfCurr1DayTimeElapsed	当前1天性能统计流逝的时间

表95（续）

性能点	描　述
adslAturPerfCurr1DayLofs	当前1天内帧丢失数
adslAturPerfCurr1DayLoss	当前1天内信号丢失数
adslAturPerfCurr1DayLprs	当前1天内电源丢失数
adslAturPerfCurr1DayESs	当前1天内错误秒丢失数
adslAturPerfPrev1DayMoniSecs	前1天性能有效的统计时间
adslAturPerfPrev1DayLofs	前1天帧丢失数
adslAturPerfPrev1DayLoss	前1天信号丢失数
adslAturPerfPrev1DayLprs	前1天电源丢失数
adslAturPerfPrev1DayESs	前1天误码丢失数
adslAtucChanReceivedBlks	收到的数据块数
adslAtucChanTransmittedBlks	发送的数据块数
adslAtucChanCorrectedBlks	纠正的数据块数
adslAtucChanUncorrectBlks	未纠正的数据块数
adslAtucChanPerfCurr15MinTimeElapsed	当前15min性能统计流逝的时间
adslAtucChanPerfCurr15MinReceivedBlks	当前15min收到的数据块数
adslAtucChanPerfCurr15MinTransmittedBlks	当前15min发送的数据块数
adslAtucChanPerfCurr15MinCorrectedBlks	当前15min纠正的数据块数
adslAtucChanPerfCurr15MinUncorrectBlks	当前15min未纠正的数据块数
adslAtucChanPerfCurr1DayTimeElapsed	当前1天性能统计流逝的时间
adslAtucChanPerfCurr1DayReceivedBlks	当前1天收到的数据块数
adslAtucChanPerfCurr1DayTransmittedBlks	当前1天发送的数据块数
adslAtucChanPerfCurr1DayCorrectedBlks	当前1天纠正的数据块数
adslAtucChanPerfCurr1DayUncorrectBlks	当前1天未纠正的数据块数
adslAtucChanPerfPrev1DayMoniSecs	前1天性能有效的统计时间
adslAtucChanPerfPrev1DayReceivedBlks	前1天收到的数据块数
adslAtucChanPerfPrev1DayTransmittedBlks	前1天发送的数据块数
adslAtucChanPerfPrev1DayCorrectedBlks	前1天纠正的数据块数
adslAtucChanPerfPrev1DayUncorrectBlks	前1天未纠正的数据块数
adslAturChanReceivedBlks	收到的数据块数
adslAturChanTransmittedBlks	发送的数据块数
adslAturChanCorrectedBlks	纠正的数据块数
adslAturChanUncorrectBlks	未纠正的数据块数
adslAturChanPerfCurr15MinTimeElapsed	当前15min性能统计流逝的时间
adslAturChanPerfCurr15MinReceivedBlks	当前15min收到的数据块数

表95（续）

性能点	描　述
adslAturChanPerfCurr15MinTransmittedBlks	当前15min发送的数据块数
adslAturChanPerfCurr15MinCorrectedBlks	当前15min纠正的数据块数
adslAturChanPerfCurr15MinUncorrectBlks	当前15min未纠正的数据块数
adslAturChanPerfCurr1DayTimeElapsed	当前1天性能统计流逝的时间
adslAturChanPerfCurr1DayReceivedBlks	当前1天收到的数据块数
adslAturChanPerfCurr1DayTransmittedBlks	当前1天发送的数据块数
adslAturChanPerfCurr1DayCorrectedBlks	当前1天纠正的数据块数
adslAturChanPerfCurr1DayUncorrectBlks	当前1天未纠正的数据块数
adslAturChanPerfPrev1DayMoniSecs	前1天性能有效的统计时间
adslAturChanPerfPrev1DayReceivedBlks	前1天收到的数据块数
adslAturChanPerfPrev1DayTransmittedBlks	前1天发送的数据块数
adslAturChanPerfPrev1DayCorrectedBlks	前1天纠正的数据块数
adslAturChanPerfPrev1DayUncorrectBlks	前1天未纠正的数据块数
ifSpeed	接口速度
ifInOctets	输入字节数
ifOutOctets	输出字节数
ifInErrors	输入错误数
ifOutErrors	输出错误数

表96　collectVdslCurrPerf接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
cardid	short，板卡标识
portid	short，端口标识
collectpoint	sequence of VdslPerfPoint，VdslPerfPoint 是 VDSL 性能点的枚举类型

输出/返回：

参　数	说　明
result	Sequence of long，性能值列表

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表97 VdslPerfPoint枚举值

性能点	描 述
vdslPhysCurrSnrMgn	VDSL当前信噪比裕度
vdslPhysCurrAtn	VDSL当前衰减
vdslPhysCurrStatus	VDSL当前状态
vdslPhysCurrOutputPwr	VDSL当前输出功率
vdslPhysCurrAttainableRate	VDSL当前可达速率
vdslPhysCurrLineRate	VDSL当前速率
vdslPerfDataLofs	帧丢失秒数
vdslPerfDataLoss	信号丢失秒数
vdslPerfDataLols	链路丢失秒数
vdslPerfDataLprs	电源丢失秒数
vdslPerfDataESs	误码秒数
vdslPerfDataSESs	严重误码秒数
vdslPerfDataUASs	不可达秒数
vdslPerfDataInits	初始化次数
vdslPerfDataCurr15MinTimeElapsed	当前15min性能统计流逝的时间
vdslPerfDataCurr15MinLofs	当前15min内帧丢失秒数
vdslPerfDataCurr15MinLoss	当前15min内信号丢失秒数
vdslPerfDataCurr15MinLols	当前15min内链路丢失秒数
vdslPerfDataCurr15MinLprs	当前15min内电源丢失秒数
vdslPerfDataCurr15MinESs	当前15min内误码丢失秒数
vdslPerfDataCurr15MinSESs	当前15min内严重误码秒数
vdslPerfDataCurr15MinUASs	当前15min内不可达秒数
vdslPerfDataCurr15MinInits	当前15min内初始化次数
vdslPerfDataCurr1DayTimeElapsed	当前1天性能统计流逝的时间
vdslPerfDataCurr1DayLofs	当前1天内帧丢失秒数
vdslPerfDataCurr1DayLoss	当前1天内信号丢失秒数
vdslPerfDataCurr1DayLols	当前1天内链路丢失秒数
vdslPerfDataCurr1DayLprs	当前1天内电源丢失秒数
vdslPerfDataCurr1DayESs	当前1天内误码丢失秒数
vdslPerfDataCurr1DaySESs	当前1天内严重误码秒数
vdslPerfDataCurr1DayUASs	当前1天内不可达秒数
vdslPerfDataCurr1DayInits	当前1天内初始化次数
vdslChanFixedOctets	纠正的字节数
vdslChanBadBlks	未纠正的数据块数

表97（续）

性能点	描　述
vdslChanCurr15MinTimeElapsed	当前15min性能统计流逝的时间
vdslChanCurr15MinFixedOctets	当前15min内纠正的字节数
vdslChanCurr15MinBadBlks	当前15min内未纠正的数据块数
vdslChanCurr1DayTimeElapsed	当前1天性能统计流逝的时间
vdslChanCurr1DayFixedOctets	当前1天内纠正的字节数
vdslChanCurr1DayBadBlks	当前1天未纠正的数据块数
VdslifSpeed	接口速度
VdslifInOctets	输入字节数
VdslifOutOctets	输出字节数
VdslifInErrors	输入错误数
VdslifOutErrors	输出错误数

表98　collectShdslCurrPerf接口操作参数

输入参数：

参　数	说　明
neid	string，网元标识
endpointindex	ShdslEndPointIndex，Shdsl端点索引
collectpoint	sequence of ShdslPerfPoint，ShdslPerfPoint 是 SHDSL 性能点的枚举类型

输出/返回：

参　数	说　明
result	Sequence of long，性能值列表

异常：

参　数	说　明
exception	NeNotExistException, IlleagalParaException

表99　ShdslEndPointIndex组成

属　性	名　称	描　述	类　型	限定符
cardid	卡标识		short	M,R
portid	端口标识		short	M,R
hdsl2ShdslInvIndex	端点标识	xtuC = 1, xtuR = 2,xru1 = 3,xru2 = 4,xru3 = 5,xru4 = 6, xru5 = 7,xru6 = 8,xru7 = 9, xru8 = 10，其中 3~10 根据可支持的中继器数目来定	short	M,R
hdsl2ShdslEndpointSide	端点侧	networkSide = 1, customerSide = 2，对 xtuc 来说，只有 customerSide，对 xtur 来说，只有 networkSide	short	M,R
hdsl2ShdslEndpointWirePair	端点线对	wirePair1 = 1 wirePair2 = 2	short	M,R

表100 ShdslPerfPoint枚举值

性能点	描 述
hdsl2ShdslEndpointCurrSnrMgn	SHDSL当前信噪比裕度
hdsl2ShdslEndpointCurrAtn	SHDSL当前衰减
hdsl2ShdslEndpointCurrStatus	SHDSL当前状态
hdsl2ShdslStatusNumAvailRepeaters	SHDSL当前中继器个数
hdsl2ShdslStatusMaxAttainableLineRate	SHDSL当前可达速率
hdsl2ShdslStatusActualLineRate	SHDSL当前速率
hdsl2ShdslStatusTransmissionModeCurrent	SHDSL当前传输模式
hdsl2ShdslEndpointES	误码秒数
hdsl2ShdslEndpointSES	严重误码秒数
hdsl2ShdslEndpointCRCanomalies	CRC检验异常数
hdsl2ShdslEndpointLOSWS	同步字丢失秒数
hdsl2ShdslEndpointUAS	不可达秒数
hdsl2ShdslEndpointCurr15MinTimeElapsed	当前15min性能统计流逝的时间
hdsl2ShdslEndpointCurr15MinES	当前15min内误码秒数
hdsl2ShdslEndpointCurr15MinSES	当前15min内严重误码秒数
hdsl2ShdslEndpointCurr15MinCRCanomalies	当前15min内CRC检验异常数
hdsl2ShdslEndpointCurr15MinLOSWS	当前15min内同步字丢失秒数
hdsl2ShdslEndpointCurr15MinUAS	当前15min内不可达秒数
hdsl2ShdslEndpointCurr1DayTimeElapsed	当前1天性能统计流逝的时间
hdsl2ShdslEndpointCurr1DayES	当前1天内误码秒数
hdsl2ShdslEndpointCurr1DaySES	当前1天内严重误码秒数
hdsl2ShdslEndpointCurr1DayCRCanomalies	当前1天内CRC检验异常数
hdsl2ShdslEndpointCurr1DayLOSWS	当前1天内同步字丢失秒数
hdsl2ShdslEndpointCurr1DayUAS	当前1天内不可达秒数
以下值只能按照端口取值，取以下的性能点时，无需指定端点相关的索引，只需确定卡标识和端口标识即可	
hdsl2ShdslifSpeed	接口速度
hdsl2ShdslifInOctets	输入字节数
hdsl2ShdslifOutOctets	输出字节数
hdsl2ShdslifInErrors	输入错误数
hdsl2ShdslifOutErrors	输出错误数

8.5 安全管理（可选）

安全管理中提供了对用户组、用户、操作权限和管理域的管理，并且提供了 EMS/NMS 间的安全认证管理。

8.5.1 接口操作表

表101 安全管理接口操作

接口名	描 述
getUserInfo	获取指定用户信息
getAllUserInfo	获取所有用户信息
getAllLog	获取所有日志
getLog	获取指定区间的日志
createUserGroup	创建一个用户组
mdUserGroup	修改一指定用户组
delGroup	删除一指定用户组
getGroup	获取所有用户组信息
createUser	创建用户
mdUserPassword	修改一指定用户的密码
delUser	删除一指定用户
getUserPerGroup	获取一指定用户组中的所有用户信息
createDomain	创建管理域
mdDomain	修改一指定管理域
delDomain	删除一指定管理域
assignDomainToUser	将一指定管理域分配给一指定用户

8.5.2 接口操作参数和结果说明

表102 getUserInfo接口操作参数

输入参数：

参 数	说 明
user	string，EMS用户标识

输出/返回：

参 数	说 明
result	UserInfo，EMS用户信息

异常：

参 数	说 明
exception	UserNotExistException

表103 UserInfo组成

属 性	名 称	描 述	类 型	限定符
username	EMS 用户标识		String	M/R
loginTime	EMS 用户登录时间		long	M/R
logoutTime	EMS 用户退出时间		long	M/R
loginCount	EMS 用户登录次数		Short	M/R

表104 getAllUserInfo接口操作参数

输入参数：无

输出/返回：

参 数	说 明
result	sequence of UserInfo，EMS的所有用户信息列表

异常：无

表105 getAllLog接口操作参数

输入参数：无

输出/返回：

参 数	说 明
result	sequence of LogInfo，EMS日志信息列表

异常：无

表106 LogInfo组成

属 性	名 称	描 述	类 型	限定符
Id	日志标识		Long	M/R
LogTime	日志记录时间		Long	M/R
user	操作 EMS 用户		String	M/R
hostAddr	操作主机地址		String	M/R
logtype	日志类型		enum LogType{normal,alarm};	M/R
devAddr	操作设备地址		String	M/R
devType	操作设备类型		String	M/R
moduletype	操作所属模块类型		String	M/R
logDescr	日志描述		String	O/R

表107 getLog接口操作参数

输入参数：

参 数	说 明
starttime	long，起始时间
endtime	long，终止时间

输出/返回：

参 数	说 明
result	Sequence of LogInfo，EMS日志信息列表

异常：

参 数	说 明
exception	IlleagalParaException

表108 mdUserGroup接口操作参数

输入参数：

参 数	说 明
group	UserGroup，用户组信息

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	UerGroupAlreadyExistException

表109 UserGroup组成

属 性	名 称	描 述	类 型	限定符
groupname	用户组名		string	M,R/W
oplist	操作权限列表	enum peration {device,faultmng,mapmng, topomng,admin};	Sequence of Operation	M,R/W

表110 createUserGroup接口操作参数

输入参数：

参 数	说 明
usergroupname	string，用户组名
oplist	sequence of Operation，操作权限列表，enum Operation {device,faultmng,mapmng,topomng,admin};

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	UserGroupNotExistException

表111 delGroup接口操作参数

输入参数：

参 数	说 明
usergroupname	string，用户组名

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	UserGroupNotExistException

表112 getGroup接口操作参数

输入参数：无

输出/返回：

参 数	说 明
result	sequence of UserGroup，EMS用户组信息列表

异常：无

表113 createUser接口操作参数

输入参数：

参 数	说 明
username	string，用户名
password	string，用户密码
list	sequence of string，用户组名列表

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	UserAlreadyExistException, UserGroupNotExistException

表114 mdUserPassword接口操作参数

输入参数：

参 数	说 明
username	string，用户名
oldpassword	string，用户旧密码
newpassword	string，用户新密码

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	UserNotExistException

表115 delUser接口操作参数

输入参数：

参 数	说 明
username	string，用户名

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	UserNotExistException

表116 getUserPerGroup接口操作参数

输入参数：

参 数	说 明
groupname	string，用户组名

输出/返回：

参 数	说 明
result	sequence of string，用户名列表

异常：

参 数	说 明
exception	UserGroupNotExistException

表117 createDomain接口操作参数

输入参数：

参 数	说 明
domainname	string，管理域名
locpathlist	sequence of string，区域列表，列表中每一个元素对应一个逻辑分组

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	DomainAlreadyExistException, NotAssignLocException

表118 mdDomain接口操作参数

输入参数：

参 数	说 明
domainname	string，管理域名
locpathlist	sequence of string，区域列表

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	omainNotExistException, NotAssignLocException

表119 delDomain接口操作参数

输入参数：

参 数	说 明
domainname	string，管理域名

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	DomainNotExistException

表120 assignDomainToUser接口操作参数

输入参数：

参 数	说 明
username	string，用户名
domainlist	sequence of string，管理域名列表

输出/返回：

参 数	说 明
result	boolean

异常：

参 数	说 明
exception	UserNotExistException, DomainNotExistException

8.6 异常定义

异常定义文件中主要定义了在上述各种管理操作中出现的异常情况。

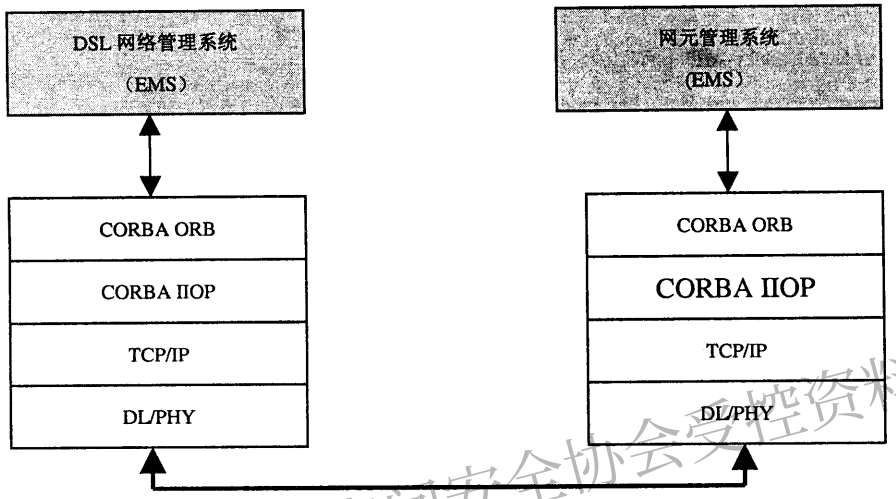
表121 异常描述

异 常	描 述
ParameterNotSupported	不支持参数
ValueNotSupported	不支持设置值
OperationNotSupported	不支持操作
UserNotExistException	用户不存在
NeNotExistException	网元不存在
IlleagalParaException	无效参数
UerGroupAlreadyExistException	用户组已经存在
UserGroupNotExistException	用户组不存在
UserAlreadyExistException	用户已经存在
DomainAlreadyExistException	管理域已经存在
NotAssignLocException	没有为创建的管理域分配逻辑分组
DomainNotExistException	管理域不存在
DomainNotAssignedException	管理域没有指定
NoRightException	没有权限
UnacknowledgeAlarms	去确认告警异常
CommentAlarms	为告警添加注释异常
GetAlarmList	同步告警列表异常
AcknowledgeAlarms	确认告警异常
GetAlarmCount	获取不同级别告警数异常

附 录 A
（资料性附录）
DSL 网络管理接口功能的 CORBA IDL 实现

A.1 CORBA接口定义说明

本规范采用CORBA做为接口方式时，接口通信协议栈采用CORBA协议栈，如下图所示。接口信息定义采用IDL语言进行描述。



实现本规范定义的接口时，CORBA ORB 及 CORBA 服务应该遵循的最低 OMG 标准版本见表 A.1。

表A.1 ORB及CORBA服务遵循的标准版本

ORB 及 CORBA 服务类别	OMG 版本号
ORB Core	2.2 或以上
命名服务	1.0 或以上
通知服务	1.0 或以上
电信日志服务(可选)	1.0 或以上（可选）
安全服务（可选）	采用“Secure IOP protocol”或“CORBA Security SSL Interoperability”（可选）
事务服务（可选）	1.1 或以上（可选）
异步消息服务（可选）	1.1 或以上(可选，由 NM 决定)

A.2 IDL接口定义文件构成

接口描述参照本文第 6 章接口管理功能需求和第 8 章接口定义的内容，将 IDL 文件分为对象管理、配置管理、告警管理、安全管理、性能管理以及操作中异常定义文件。

A.3 IDL接口定义

A.3.1 公共管理

公共管理包括对象管理和通知等通用管理。

通用管理：comm.idl

```
#include <error.idl>
module comm{
    //just a empty interface for inherience
    interface CommService{
    };

    interface ServiceMng{
        CommService getSpecialService(in string servtype,in string netype,in string
            businesstype) raises (error::NoRightException);
        CommService getCommService(in string servtype,in string netype)
            raises (error::NoRightException);
    };
    interface NotificationMng:comm::CommService{
        unsigned long notreg(in string pushconsumercorbaloc);
        void disNotReg(in unsigned long nmsconsumerid);
    };
    interface LoginManager {
        /* NMS 提供用户名和密码向 EMS 系统请求认证。
        ServiceMng loginEms (in string username,in string password)
            raises(error::UserNotExistException);
    };
};
```

对象管理：basicmng.idl,dslmng.idl

```
#include <error.idl>
#include <comm.idl>
module basicmng{
    struct EmsSysInfo{
        string emsname;
        unsigned long long starttime;
        string emsipaddr;
        wstring emslocation;
        string emsversion;
        string emsvendor;
        string emshardwareprovider;
        string emdhardwaretype;
        string emsopersys;
```

```
        string emsdatabasetype;
};
interface Top{
        const string CLASS = "Top";
        const string ID = "Id";
        const string NAME = "Name";
        const string PARENT =    "Parent";
        const string ISLEAF = "isLeaf";
};
interface NetWorkVirtualRoot:Top{
        const string CLASS = "NetWorkVirtualRoot";
};
interface Group:Top{
        const string CLASS = "Group";
};
interface Ne:Top{
        const string CLASS = "Ne";
        const string NEIPADDR = "Neipaddr";
        const string NETYPE = "Netype";
        const string DN = "Dn";
};
typedef string MOAttributeName;
typedef any MOAttributeValue;
struct MOAttribute{
        MOAttributeName name;
        MOAttributeValue value;
};

typedef string MOCategory;
typedef sequence<MOAttribute>    MOAttributeList;
struct MO{
        MOCategory category;
        MOAttributeList attributes;
};
typedef sequence<MO> MOList;
//return the tree in mid-order
interface TopoMng:comm::CommService{
```

```
            EmsSysInfo getEmssys();
            MOList getTopoTree();
        };
    };

    #include <comm.idl>
    module dslmng{
        enum CardTypeDef{
            CARD_WANIP,
            CARD_WANATM,
            CARD_ADSL,
            CARD_SHDSL,
            CARD_VDSL,
            CARD_IMA,
            CARD_E1,
            CARD_LAN,
            CARD_CORE,
            CARD_OTHER
        };
        struct CardInfo{
            short cardno;
            CardTypeDef cardtype;
            string cardtypedesc;
            short portnum;
            string hardwareversion;
            string softwareversion;
            short cpuload;
            short memusage;
        };
        const short NotSupport = -1;
        typedef sequence<CardInfo> CardInfoList;
        struct NeInfo{
            string ipaddr;
            string inbandmac;
            string outbandmac;
            string netype;
            wstring label;
```

```
            string version;
            string vendorname;
            string locationInfo;
            short slotnum;
            CardInfoList cardInfo;
        };
        typedef sequence<NeInfo> NeInfoList;
        typedef sequence<string> NeipList;
        interface TopoMng:framework::comm::CommService{
          NeInfoList getNeTopo(in NeipList neips) raises(framework::error::NeNotExistException);
        };
    };
```

A.3.2 配置管理

confmng.idl：

```
    #include <error.idl>
    #include <comm.idl>
    module conf{
        struct ReadOnlyNeConf{
            string sysDescr;
            string sysObjectID;
            unsigned long long sysUpTime;
        };
        struct CanSetNeConf{
            string sysName;
            string sysLocation;
            string sysContact;
        };
        struct NeConf{
            ReadOnlyNeConf readattr;
            CanSetNeConf    setattr;
        };
        enum IfStatusType{
            up,
            down,
            testing
        };
```

```
typedef IfStatusType IfOperStatusType;
struct ReadOnlyAdslLineConf{
    IfOperStatusType ifOperStatus;
    unsigned long adslAturChanCurrTxRate;
    unsigned long adslAtucChanCurrTxRate;
};
enum LineCoding{
    other,              //1
    dmt,                //2
    cap,                //3
    qam,                //4
    glite               //5
};
enum LineType{
    noChannel,          //1
    fastOnly,           //(2),
    interleavedOnly,    // (3),
    fastOrInterleaved,  // (4),
    fastAndInterleaved  // (5)
};
typedef IfStatusType IfAdminStatusType;
struct CanSetAdslLineConf{
    IfAdminStatusType ifAdminStatus;
    LineCoding adslLineCoding;
    LineType adslLineType;
    string adslLineConfProfile;
    string adslLineAlarmConfProfile;
    string username;
    string userphone;
};
struct AdslLineConf{
    string neipaddr;
    short cardno;
    short portno;
    ReadOnlyAdslLineConf readattr;
    CanSetAdslLineConf setattr;
};
```

```
enum RateMode{
    fixedmode,
    adaptAtStartup,
    adaptAtRuntime
};
struct AdslLineConfProfileTable{
    string   adslLineConfProfileName;      // index, 32 byte

    /**
     * Atuc 线路速率模式，取值范围：<br>
     * fixed = 1,              // no rate adaptation<br>
     * adaptAtStartup = 2,     // perform rate adaptation only at initialization<br>
     * adaptAtRuntime = 3      // perform rate adaptation at any time
     * */
    RateMode                adslAtucConfRateMode;
    short                   adslAtucConfRateChanRatio;
    short                   adslAtucConfTargetSnrMgn;
    short                   adslAtucConfMaxSnrMgn;
    short                   adslAtucConfMinSnrMgn;
    short                   adslAtucConfDownshiftSnrMgn;
    short                   adslAtucConfUpshiftSnrMgn;
    short                   adslAtucConfMinUpshiftTime;
    short                   adslAtucConfMinDownshiftTime;
    unsigned long long      adslAtucChanConfFastMinTxRate;
    unsigned long long      adslAtucChanConfInterleaveMinTxRate;
    unsigned long long      adslAtucChanConfFastMaxTxRate;
    unsigned long long      adslAtucChanConfInterleaveMaxTxRate;
    short                   adslAtucChanConfMaxInterleaveDelay;

    /**
     * Atur 线路速率模式，取值范围：<br>
     * fixed = 1,              // no rate adaptation<br>
     * adaptAtStartup = 2,     // perform rate adaptation only at initialization<br>
     * adaptAtRuntime = 3      // perform rate adaptation at any time
     * */
     RateMode                   adslAturConfRateMode;
     short                      adslAturConfRateChanRatio;
```

```
        short                   adslAturConfTargetSnrMgn;
        short                   adslAturConfMaxSnrMgn;
        short                   adslAturConfMinSnrMgn;
        short                   adslAturConfDownshiftSnrMgn;
        short                   adslAturConfUpshiftSnrMgn;
        short                   adslAturConfMinUpshiftTime;
        short                   adslAturConfMinDownshiftTime;
        unsigned long long      adslAturChanConfFastMinTxRate;
        unsigned long long      adslAturChanConfInterleaveMinTxRate;
        unsigned long long      adslAturChanConfFastMaxTxRate;
        unsigned long long      adslAturChanConfInterleaveMaxTxRate;
        short                   adslAturChanConfMaxInterleaveDelay;
};
struct AdslLineAlarmConfProfileTable{
    string                  adslLineAlarmConfProfileName;
    short                   adslAtucThresh15MinLofs;
    short                   adslAtucThresh15MinLoss;
    short                   adslAtucThresh15MinLols;
    short                   adslAtucThresh15MinLprs;
    short                   adslAtucThresh15MinESs;
    unsigned long long      adslAtucThreshFastRateUp ;
    unsigned long long      adslAtucThreshInterleaveRateUp;
    unsigned long long      adslAtucThreshFastRateDown;
    unsigned long long      adslAtucThreshInterleaveRateDown;
    short                   adslAtucInitFailureTrapEnable;
    short                   adslAturThresh15MinLofs;
    short                   adslAturThresh15MinLoss;
    short                   adslAturThresh15MinLprs;
    short                   adslAturThresh15MinESs;
    unsigned long long      adslAturThreshFastRateUp;
    unsigned long long      adslAturThreshInterleaveRateUp;
    unsigned long long      adslAturThreshFastRateDown;
    unsigned long long      adslAturThreshInterleaveRateDown;
};
//adsl2+ definition
enum TransMode{
    ansit1413,                  //(0)
```

```
    etsi,                                   //(1)
    q9921PotsNonOverlapped,                 //(2)
    q9921PotsOverlapped,                    //(3)
    q9921IsdnNonOverlapped,                 //(4)
    q9921isdnOverlapped,                    //5
    q9921tcmIsdnNonOverlapped,              //(6)
    q9921tcmIsdnOverlapped,                 //7
    q9922potsNonOverlapeed,                 //(8)
    q9922potsOverlapped,                    //(9)
    q9922tcmIsdnNonOverlapped,              //(10)
    q9922tcmIsdnOverlapped,                 //(11)
    q9921tcmIsdnSymmetric                   //(12)
};
typedef sequence<TransMode> TransModeList;
struct Adsl2PlusLineConf{
    IfOperStatusType ifOperStatus;
    IfAdminStatusType ifAdminStatus;
    TransModeList adslLineTransAtucConfig;   //config
    TransModeList adslLineTransAtucCap;      //support
    TransModeList adslLineTransAtucActual;   //actual
    string adslLineConfProfile;
    string adslLineAlarmConfProfile;
    wstring username;
    string userphone;
};

struct Adsl2PlusLineConfProfileTable{
    AdslLineConfProfileTable adsllineprf;
    LineType adslConfProfileLineType;
};

struct Adsl2PlusLineAlarmConfProfileTable{
    AdslLineAlarmConfProfileTable adslalarmprf;
    short adslAtucThreshold15MinFailedFastR;     //INTEGER(0..900), "seconds"
    short adslAtucThreshold15MinSesL;            //INTEGER(0..900), "seconds"
    short adslAtucThreshold15MinUasL;            //INTEGER(0..900), "seconds"
    short adslAturThreshold15MinSesL;            //INTEGER(0..900), "seconds"
```

```
    short adslAturThreshold15MinUasL;              //INTEGER(0..900), "seconds"
};

//vdsl definition
enum VdslLineCoding{
    othercoding,                    // 1,none of the following
    mcm,                            //2, Multiple Carrier Modulation
    scm                             // 3,Single Carrier Modulation
};

struct VdslLineConf{
    IfAdminStatusType ifAdminStatus;
    IfOperStatusType ifOperStatus;
    VdslLineCoding      lineCoding;
    LineType            vdslLineType;
    string              vdslLineConfProfile;
    string              vdslLineAlarmConfProfile;
    wstring username;
    string userphone;
};

enum VdslRateMode{
    manualRateMode,                 //1, forces the rate to the configured rate
    adaptAtInit                     //2, adapts the line based upon line quality
};

enum VdslPboControl{
    disabled,                   //1,do not support downstream PBO control
    auto,                       //2,automatically adjust the power backoff
    manualPboControl            //3,use the value from vdslLineConfDownPboLevel
};
struct VdslLineConfProfileTable{
    string  vdslLineConfProfileName;    // index, 32 byte
    /**
    * 下行速率模式，取值范围：<br>
    * manual = 1,       // forces the rate to the configured rate<br>
    * adaptAtInit = 2   // adapts the line based upon line quality<br>
```

```
    * */
    VdslRateMode        vdslLineConfDownRateMode;
    VdslRateMode        vdslLineConfUpRateMode;
    short               vdslLineConfDownMaxPwr;          // (0..58), "0.25dBm"
    short               vdslLineConfUpMaxPwr;            // (0..58), "0.25dBm"
    short               vdslLineConfDownMaxSnrMgn;       // (0..127), "0.25dB"
    short               vdslLineConfDownMinSnrMgn;       // (0..127), "0.25dB"
    short               vdslLineConfDownTargetSnrMgn;    // (0..127), "0.25dB"
    short               vdslLineConfUpMaxSnrMgn;         // (0..127), "0.25dB"
    short               vdslLineConfUpMinSnrMgn;         // (0..127), "0.25dB"
    short               vdslLineConfUpTargetSnrMgn;      // (0..127), "0.25dB"
    unsigned long long  vdslLineConfDownFastMaxDataRate; // "kbit/s"
    unsigned long long  vdslLineConfDownFastMinDataRate; // "kbit/s"
    unsigned long long  vdslLineConfDownSlowMaxDataRate; // "kbit/s"
    unsigned long long  vdslLineConfDownSlowMinDataRate; // "kbit/s"
    unsigned long long  vdslLineConfUpFastMaxDataRate;   // "kbit/s"
    unsigned long long  vdslLineConfUpFastMinDataRate;   // "kbit/s"
    unsigned long long  vdslLineConfUpSlowMaxDataRate;   // "kbit/s"
    unsigned long long  vdslLineConfUpSlowMinDataRate;   // "kbit/s"
    short               vdslLineConfDownRateRatio;       // (0..100), "%"
    short               vdslLineConfUpRateRatio;         // (0..100), "%"
    short               vdslLineConfDownMaxInterDelay;   // (0..255), " milli-seconds"
    short               vdslLineConfUpMaxInterDelay;     // (0..255), " milli-seconds"
    VdslPboControl      vdslLineConfDownPboControl;
    VdslPboControl      vdslLineConfUpPboControl;        ""
    short               vdslLineConfDownPboLevel;        // (0..160), ""
    short               vdslLineConfUpPboLevel;          // (0..160), ""
};

struct VdslLineAlarmConfProfileTable{
    string    vdslLineAlarmConfProfileName;              // index, 32 bytes
    short     vdslLineAlarmConfThresh15MinLofs;          // (0..900), "seconds"
    short     vdslLineAlarmConfThresh15MinLoss;          // (0..900), "seconds"
    short     vdslLineAlarmConfThresh15MinLprs;          // (0..900), "seconds"
    short     vdslLineAlarmConfThresh15MinLols;          // (0..900), "seconds"
    short     vdslLineAlarmConfThresh15MinESs;           // (0..900), "seconds"
    short     vdslLineAlarmConfThresh15MinSESs;          // (0..900), "seconds"
```

```
    short    dslLineAlarmConfThresh15MinUASs;          // (0..900), "seconds"
    short    dslLineAlarmConfInitFailure;              // true: enable(1); false: disable(2)
};

//shdsl definition
struct Hdsl2ShdslSpanConfTable{
    IfAdminStatusType ifAdminStatus;
    IfOperStatusType ifOperStatus;                     //只读
    short  hdsl2ShdslSpanConfNumRepeaters;     // (0~8), 可以配置的中继器数目
    string                  hdsl2ShdslSpanConfProfile;
    string                  hdsl2ShdslSpanConfAlarmProfile;
    wstring username;
    string userphone;
    short  hdsl2ShdslStatusNumAvailRepeaters;  //自动发现的中继器数目(只读)
};
struct Hdsl2ShdslEndpointConfTable{
    short cardno;
    short portno;

    /**
    * 端点，其中 STUC 和 STUR 是肯定有的，Repeater1~8 有几个需要根据
    *   hdsl2ShdslStatusNumAvailRepeaters 判断，取值范围：<br>
    * xtuC = 1,          // STUC <br>
    * xtuR = 2,          // STUR <br>
    * xru1 = 3,          // Repeater1<br>
    * xru2 = 4,          // Repeater2<br>
    * xru3 = 5,          // Repeater3<br>
    * xru4 = 6,          // Repeater4 <br>
    * xru5 = 7,          // Repeater5<br>
    * xru6 = 8,          // Repeater6<br>
    * xru7 = 9,          // Repeater7<br>
    * xru8 = 10          // Repeater8<br>
    * */
    short                hdsl2ShdslInvIndex;        // index
    /**
    *端点侧，取值范围：<br>
    * networkSide = 1,        // on the Network side <br>
```

```
    * customerSide = 2        // on the Customer side <br>
    * */
    //对 xtuc 来说，只有 customerSide，对 xtur 来说，只有 networkSide
    short                hdsl2ShdslEndpointSide;          // index
    /**
    *端点线对，取值范围：<br>
    * wirePair1 = 1,          // the first pair <br>
    * wirePair2 = 2           // the optional second pair <br>
    * */
    short                hdsl2ShdslEndpointWirePair;          // index
    //端点配置中只有此量可以设置，其余均为索引
    string   hdsl2ShdslEndpointAlarmConfProfile;
};
enum ShdslSpanConfTransmissionMode{
    Region1,                 //bit(0)
    Region2                  //bit(1)
};
typedef   sequence<ShdslSpanConfTransmissionMode>
          ShdslSpanConfTransmissionModes;
enum ShdslSpanConfUsedTargetMargin{
    currCondDown,                    //第 0 位
    worstCaseDown,                   //第 1 位
    currCondUp,                      //第 2 位
    worstCaseUp                      //第 3 位
};
typedef   sequence<ShdslSpanConfUsedTargetMargin>
          ShdslSpanConfUsedTargetMargins;
struct ShdslLineConfProfileTable{
    string   hdsl2ShdslSpanConfProfileName;      // index, 32 byte
    /**
    * 线制，取值范围：<br>
    * TwoWire = 1,          // two-wire<br>
    * fourWire = 2          // four-wire <br>
    * */
    short      hdsl2ShdslSpanConfWireInterface;              // (1..2), ""
    unsigned long long hdsl2ShdslSpanConfMinLineRate;      // (0..4112000), "bit/s"
    unsigned long long hdsl2ShdslSpanConfMaxLineRate;      // (0..4112000), "bit/s"
```

```
/**
* 功率谱密度，取值范围：<br>
* symmetric = 1,          // symmetric <br>
* asymmetric = 2    // asymmetric <br>
* */
short       hdsl2ShdslSpanConfPSD;                  // (1..2), ""

//按位或的值，序列的长度不超过 2
ShdslSpanConfTransmissionModes      hdsl2ShdslSpanConfTransmissionMode;

/**
* 远程管理使能，取值范围：<br>
* enable = 1,          // enable <br>
* disable = 2          // disable <br>
* */
short     hdsl2ShdslSpanConfRemoteEnabled;                  // (1..2), ""
/**
* 电源馈电，取值范围：<br>
* noPower = 1,          // noPower <br>
* powerFeed = 2,        // powerFeed <br>
* wettingCurrent = 3      // wettingCurrent <br>
* */
short       hdsl2ShdslSpanConfPowerFeeding;                  // (1..3), ""
long hdsl2ShdslSpanConfCurrCondTargetMarginDown;      // (－10..21), "dB"
long hdsl2ShdslSpanConfWorstCaseTargetMarginDown;    // (－10..21), "dB"
long hdsl2ShdslSpanConfCurrCondTargetMarginUp;        // (－10..21), "dB"
long hdsl2ShdslSpanConfWorstCaseTargetMarginUp;       // (－10..21), "dB"
//按位或的值，序列的长度不超过 4
ShdslSpanConfUsedTargetMargins   hdsl2ShdslSpanConfUsedTargetMargins;
/**
* 参考时钟类型，取值范围：<br>
* localClk = 1,                  // Mode-1 per G991.2<br>
* networkClk = 2,                // Mode-2 per G991.2<br>
* dataOrNetworkClk = 3,          // Mode-3a per G991.2<br>
* dataClk = 4                    // Mode-3b per G991.2<br>
* */
short          hdsl2ShdslSpanConfReferenceClock;                  // (1..4), ""
```

```
    /**
    * 线路探测使能，取值范围：<br>
    * enable = 1,          // enable <br>
    * disable = 2          // disable <br>
    * */
    short     hdsl2ShdslSpanConfLineProbeEnable;              // (1..2), ""
};
struct ShdslLineAlarmConfProfileTable{
    string     hdsl2ShdslEndpointAlarmConfProfileName;   // index, 32 bytes
    long    dsl2ShdslEndpointThreshLoopAttenuation;      // (－127 .. 128), "0.1dB"
    long    hdsl2ShdslEndpointThreshSNRMargin;           // (－127 .. 128), "0.1dB"
    short    dsl2ShdslEndpointThreshES;                  // (0..900), "seconds"
    short     hdsl2ShdslEndpointThreshSES;               // (0..900), "seconds"
    short     hdsl2ShdslEndpointThreshCRCanomalies;      // (0..900), "seconds"
    short     hdsl2ShdslEndpointThreshLOSWS;             // (0..900), "seconds"
   short     hdsl2ShdslEndpointThreshUAS;                // (0..900), "seconds"
                        };
interface ConfigMng:: Comm: CommService {
    //获取指定网元的相关属性
    NeConf getNeConf(in string neid)
            raises(error::NeNotExistException);
    //设置指定网元的相关属性
    boolean setNeConf(in string neid,in CanSetNeConf conf)
            raises (error::NeNotExistException);
    //获取指定 ADSL 线路的相关配置信息
    AdslLineConf getAdslLineConf (in string neid,in short cardno,in short portno)
            raises (error::NeNotExistException,error::IlleagalParaException) ;
    //设置指定 ADSL 线路的相关配置信息
    boolean setAdslLineConf (in string neid,in short cardno,in short portno,in CanSetAdslLineConf
        conf)
            raises (error::NeNotExistException,error::IlleagalParaException);
     //获取指定 ADSL 线路参数模板信息
    AdslLineConfProfileTable getAdslLineConfProfile(in string neipaddr,in string
        lineprofilename)
            raises (error::NeNotExistException,error::IlleagalParaException);
    //创建 ADSL 线路参数模板
    boolean crAdslLineConfProfile(in string neid,in AdslLineConfProfileTable lineconf) raises
```

```
        (error::NeNotExistException,error::IlleagalParaException) ;
    //修改指定 ADSL 线路参数模板
    boolean mdAdslLineConfProfile(in string neid,in AdslLineConfProfileTable lineconf) raises
        (error::NeNotExistException,error::IlleagalParaException) ;
    //删除指定 ADSL 线路参数模板
    boolean delAdslLineConfProfile(in string neid,in string adslLineConfProfileName) raises
        (error::NeNotExistException) ;
    //获取指定 ADSL 告警参数模板信息
    AdslLineAlarmConfProfileTable getAdslAlarmProfile(in string neipaddr,in string
        alarmprofilename)
          raises (error::NeNotExistException,error::IlleagalParaException);
    //创建 ADSL 告警参数模板
    boolean crAdslLineAlarmConfProfile(in string neid,in AdslLineAlarmConfProfileTable
      alarmconf)
          raises (error::NeNotExistException,error::IlleagalParaException);
    //修改指定 ADSL 告警参数模板
    boolean mdAdslLineAlarmConfProfile(in string neid,in AdslLineAlarmConfProfileTable
      alarmconf)
          raises (error::NeNotExistException,error::IlleagalParaException);
    //删除指定 ADSL 告警参数模板
    boolean delAdslLineAlarmConfProfile(in string neid,in string adslLineAlarmConfProfileName)
           raises (error::NeNotExistException) ;
    //ADSL2+
    //获取 ADSL2plus 线路配置
    Adsl2PlusLineConf getAdslLine2PlusConf(in string neipaddr,in short cardno,in
        short portno)
          raises (error::NeNotExistException,error::IlleagalParaException) ;
    //Adsl2plus 线路配置
    boolean setAdslLine2PlusConf(in string neipaddr,in short cardno,in short
        portno,inout Adsl2PlusLineConf conf)
           raises (error::NeNotExistException,error::IlleagalParaException);
    //获取 Adsl2plus 线路配置参数模板信息
    Adsl2PlusLineConfProfileTable getAdsl2PlusLineConfProfile(in string neipaddr,in
      string lineprofilename)
          raises (error::NeNotExistException,error::IlleagalParaException);
    //创建 Adsl2plus 线路配置参数模板
    boolean crAdsl2PlusLineConfProfile(in string neipaddr,in
```

```
    Adsl2PlusLineConfProfileTable lineconf)
        raises (error::NeNotExistException,error::IlleagalParaException) ;
//修改 Adsl2plus 线路配置参数模板
boolean mdAdsl2PlusLineConfProfile(in string neipaddr,in
    Adsl2PlusLineConfProfileTable lineconf)
        raises (error::NeNotExistException,error::IlleagalParaException) ;
//删除 Adsl2plus 线路配置参数模板
boolean delAdsl2PlusLineConfProfile(in string neipaddr,in string
    adslLineConfProfileName)
      raises (error::NeNotExistException) ;
//获取 ADSL2plus 告警参数模板信息
Adsl2PlusLineAlarmConfProfileTable getAdsl2PlusAlarmProfile(in string
    neipaddr,in string alarmprofilename)
        raises (error::NeNotExistException,error::IlleagalParaException);
//创建 Adsl2plus 告警参数模板
boolean crAdsl2PlusLineAlarmConfProfile(in string neipaddr,in
    Adsl2PlusLineAlarmConfProfileTable alarmconf)
        raises (error::NeNotExistException,error::IlleagalParaException);
//修改 Adsl2plus 告警参数模板
boolean mdAdsl2PlusLineAlarmConfProfile(in string neipaddr,in
    Adsl2PlusLineAlarmConfProfileTable alarmconf)
        raises (error::NeNotExistException,error::IlleagalParaException);
//删除 Adsl2plus 告警参数模板
boolean delAdsl2PlusLineAlarmConfProfile(in string neipaddr,in
    string adslLineAlarmConfProfileName) raises (error::NeNotExistException) ;

//VDSL
VdslLineConf getVdslLineConf(in string neipaddr,in short cardno,in short portno)
    raises (error::NeNotExistException,error::IlleagalParaException) ;
boolean setVdslLineConf(in string neipaddr,in short cardno,in short portno, in
VdslLineConf conf)
raises (error::NeNotExistException,error::IlleagalParaException);
  VdslLineConfProfileTable getVdslLineConfProfile(in string neipaddr,in string
lineprofilename) raises
      (error::NeNotExistException,error::IlleagalParaException);
boolean crVdslLineConfProfile(in string neipaddr,in VdslLineConfProfileTable
  lineconf) raises (error::NeNotExistException,error::IlleagalParaException) ;
```

```
boolean mdVdslLineConfProfile(in string neipaddr,in VdslLineConfProfileTable
    lineconf) raises (error::NeNotExistException,error::IlleagalParaException) ;
boolean delVdslLineConfProfile(in string neipaddr,in string lineprofilename) raises
  (error::NeNotExistException) ;
VdslLineAlarmConfProfileTable getVdslAlarmProfile(in string neipaddr,in string
 alarmprofilename)
       raises (error::NeNotExistException,error::IlleagalParaException);
boolean crVdslLineAlarmConfProfile(in string neipaddr,in
     VdslLineAlarmConfProfileTable alarmconf)
     raises (error::NeNotExistException,error::IlleagalParaException);
boolean mdVdslLineAlarmConfProfile(in string neipaddr,in
     VdslLineAlarmConfProfileTable alarmconf)
     raises (error::NeNotExistException,error::IlleagalParaException);
boolean delVdslLineAlarmConfProfile(in string neipaddr,in string
     alarmprofilename) raises (error::NeNotExistException) ;

        //SHDSL
        Hdsl2ShdslSpanConfTable getShdslLineConf(in string neipaddr,in short cardno,in
     short portno)
      raises (error::NeNotExistException,error::IlleagalParaException) ;
boolean setShdslLineConf(in string neipaddr,in short cardno,in short portno,in
    Hdsl2ShdslSpanConfTable conf)
    raises (error::NeNotExistException,error::IlleagalParaException);
//shdsl 端点配置
void getHdsl2ShdslEndpoint(in string neipaddr, inout
       Hdsl2ShdslEndpointConfTable pointconf)
       raises (error::NeNotExistException,error::IlleagalParaException) ;
 boolean setHdsl2ShdslEndpoint(in string neipaddr, in
        Hdsl2ShdslEndpointConfTable pointconf)
        raises (error::NeNotExistException,error::IlleagalParaException);
          ShdslLineConfProfileTable getShdslLineConfProfile(in string neipaddr,in string
       lineprofilename)
        raises (error::NeNotExistException,error::IlleagalParaException);
boolean crShdslLineConfProfile(in string neipaddr,in ShdslLineConfProfileTable
   lineconf) raises (error::NeNotExistException,error::IlleagalParaException) ;
boolean mdShdslLineConfProfile(in string neipaddr,in ShdslLineConfProfileTable
    lineconf) raises (error::NeNotExistException,error::IlleagalParaException) ;
```

```
        boolean delShdslLineConfProfile(in string neipaddr,in string lineprofilename)
            raises (error::NeNotExistException) ;
              ShdslLineAlarmConfProfileTable getShdslAlarmProfile(in string neipaddr,in string
             alarmprofilename)
              raises (error::NeNotExistException,error::IlleagalParaException);
        boolean crShdslLineAlarmConfProfile(in string neipaddr,in
              ShdslLineAlarmConfProfileTable alarmconf)
            raises (error::NeNotExistException,error::IlleagalParaException);
        boolean mdShdslLineAlarmConfProfile(in string neipaddr,in
              ShdslLineAlarmConfProfileTable alarmconf)
              raises (error::NeNotExistException,error::IlleagalParaException);
        boolean delShdslLineAlarmConfProfile(in string neipaddr,in string
              alarmprofilename) raises (error::NeNotExistException) ;
        //禁用指定端口
        boolean disablePort(in string neid,in short cardno,in short portno)
                raises (error::NeNotExistException,error::IlleagalParaException) ;
        //启用指定端口
        boolean enablePort(in string neid,in short cardno,in short portno)
                raises (error::NeNotExistException,error::IlleagalParaException) ;
        //复位指定端口
        boolean resetPort(in string neid,in short cardno,in short portno)
                raises (error::NeNotExistException,error::IlleagalParaException) ;
    };
};
```

A.3.3 告警管理

告警相关常量定义文件：alarmconsetdef.idl

```
#ifndef _AlarmConstDefs_idl_
#define _AlarmConstDefs_idl_

#include " CosNotification.idl"
#include "commconstdef.idl"


/* ## Module: alarmconstdef
This module contains commonly used definitions for Alarm IRP
==========================================================
```

```
*/
module alarm{
    module constdef{
        /*
        This block identifies the alarm types specified for this IRP version.
        These types carry the same semantics as the TMN ITU-T defined event types of the
        same name.Their encodings for this version of Alarm IRP are defined here.  Other
        IRP documents, or other versions of Alarm IRP, shall identify their own alarm types
        for their use. They shall define their encodings as well.  Values defined here are
        unique among themselves.
        */
        interface AlarmType
        {
            const string COMMUNICATIONS_ALARM = "x1";
            const string PROCESSING_ERROR_ALARM = "x2";
            const string ENVIRONMENTAL_ALARM = "x3";
            const string QUALITY_OF_SERVICE_ALARM = "x4";
            const string EQUIPMENT_ALARM = "x5";

        };
        /*
        This block identifies the notification types defined by this
        Alarm IRP version.
        */
        interface NotificationType
        {
            const string NOTIFY_FM_NEW_ALARM = "x1";
            const string NOTIFY_FM_CHANGED_ALARM = "x2";
            const string NOTIFY_FM_ACK_STATE_CHANGED = "x3";
            const string NOTIFY_FM_COMMENT_ADDED = "x4";
            const string NOTIFY_FM_CLEARED_ALARM = "x5";
         };

         /*
         This block identifies the levels of severity.
         */
         interface PerceivedSeverity
```

```
{
    const short INDETERMINATE = 1;
    const short CRITICAL = 2;
    const short MAJOR = 3;
    const short MINOR = 4;
    const short WARNING = 5;
    const short CLEARED = 6;
};
/*
Define the structure of Alarm ID and Perceived Severity used within the alarm
acknowledgment operation. Note: perceived_severity is an optional parameter.
If this value is present, it must have one of the defined values of Interface
PerceivedSeverity.
*/
struct AlarmInformationIdAndSev
{
    string alarm_information_reference;
    comm::constdef::ShortTypeOpt perceived_severity;
};

/*
Define set of the above structure of Alarm ID and Perceived Severity.
*/
typedef sequence <AlarmInformationIdAndSev> AlarmInformationIdAndSevSeq;

/*
It indicates the reason for an alarm acknowledgement to have failed:
- The specified Alarm Information is absent from the Alarm List
- The Perceived Severity to be acknowledged has changed and/or is different within
 the Alarm List - The acknowledgement failed for some other reason
*/
enum AcknowledgeFailureCategories
{
    UNKNOWNALARMID,
    WRONGPERCEIVEDSEVERITY,
    ACKNOWLEDGMENTFAILED
};
```

```
/*
Define the structure returned when the acknowledge operation fails for a set of
alarm ids. A failure category and a reason are provided in order to indicate why the
 operation failed.
*/
struct BadAcknowledgeAlarmInfo
{
    string alarm_information_reference;
    AcknowledgeFailureCategories failure_category;
    string reason;
};
typedef sequence <BadAcknowledgeAlarmInfo> BadAcknowledgeAlarmInfoSeq;
typedef sequence <string> AlarmInformationIdSeq;
/*
Define the structure returned when an operation fails for a set of alarm ids.A
reason is provided in order to indicate why the operation failed.
*/
struct BadAlarmInformationId
{
    string alarm_information_reference;
    string reason;
};

typedef sequence <BadAlarmInformationId> BadAlarmInformationIdSeq;
typedef CosNotification::EventBatch AlarmInformationSeq;
interface AttributeNameValue
{
    //common filter body
    const string NETYPE = "b";
    const string NEIP = "c";
    const string EVENT_TIME = "e";
    const string PROBABLE_CAUSE = "g";
    const string PERCEIVED_SEVERITY = "h";
    //common remain body
    const string ENTITY_ID = "j";
    const string ALARM_ID = "f";
```

```
            const string NOTIFICATION_ID = "d";
            //clear alarm remain body
            const string CLEAR_USER_ID = "y";
            const string CLEAR_SYSTEM_ID = "z";
            //ackstatechanged remain body
            const string ACK_USER_ID = "l";
            const string ACK_SYSTEM_ID = "m";
            const string ACK_STATE = "n";
            //comment remain body
            const string COMMENTS = "o";
            //list alarm
            const string ALARM_CLEARED_TIME = "ll";
            const string ALARM_CHANGED_TIME = "mm";
            const string ACK_TIME = "k";
        };
        /*
        This block identifies the acknowledgement state of a reported alarm.
        */
        interface AckState
        {
            const short ACKNOWLEDGED = 1;
            const short UNACKNOWLEDGED = 2;
        };
        Enum DslProbCause{ColdStart,
            AuthenticationFailure,LinkDown,
            LinkUp,AdslAtucTraps,AdslAturTraps,
            CardUp ,CardDown,FanOpenStatusChange,
            PowerStatusChange,NeAdded,NeDeleted,
            SplitterCardUp,SplitterCardDown
        };
    };
};
#endif //_AlarmConstDefs_idl_
```

告警管理（alarmmng.idl）

```
#ifndef _Alarm_idl_
#define _Alarm_idl_
```

```
#include <comm.idl>
#include <alarmconstdef.idl>
#include <commconstdef.idl>
#include <error.idl>
module alarm{
    /*
    System fails to complete the operation.  System can provide reason
    to qualify the exception.  The semantics carried in reason
    is outside the scope of this IRP.
    */
    exception AcknowledgeAlarms { string reason; };
    exception UnacknowledgeAlarms { string reason; };
    exception CommentAlarms { string reason; };
    exception ClearAlarms { string reason; };
    exception GetAlarmList { string reason; };
    exception GetAlarmCount { string reason; };
    exception NextAlarmInformations { string reason; };
    /*
    The AlarmInformationIterator is used to iterate through a snapshot of
    Alarm Informations taken from the Alarm List when IRPManager invokes
    get_alarm_list. IRPManager uses it to pace the return of Alarm
    Informations.IRPAgent controls the life-cycle of the iterator. However, a destroy
    operation is provided to handle the case where IRPManager wants to stop the iteration
    procedure before reaching the last iteration.
    */
    typedef comm::constdef::Signal Signal;
    typedef comm::constdef::StringTypeOpt StringTypeOpt;
    typedef alarm::constdef::AlarmInformationSeq AlarmInformationSeq;
    typedef alarm::constdef::AlarmInformationIdSeq AlarmInformationIdSeq;
    typedef   alarm::constdef::AlarmInformationIdAndSevSeq
              AlarmInformationIdAndSevSeq;
    typedef   alarm::constdef::BadAcknowledgeAlarmInfoSeq
              BadAcknowledgeAlarmInfoSeq;
    typedef alarm::constdef::BadAlarmInformationIdSeq BadAlarmInformationIdSeq;

    interface AlarmMng:comm::CommService {
        /*
```

```
Request to acknowledge one or more alarms.
*/
Signal acknowledge_alarms (
    in AlarmInformationIdSeq
          alarm_information_id_list,
    in string ack_user_id,
    in StringTypeOpt ack_system_id,
    out BadAcknowledgeAlarmInfoSeq
    bad_ack_alarm_info_list
 )  raises (AcknowledgeAlarms, error::ParameterNotSupported,
                                  error::IlleagalParaException);

/*
Request to remove acknowledgement information of one or more alarms.
*/
Signal unacknowledge_alarms (
     in AlarmInformationIdSeq alarm_information_id_list,
     in string ack_user_id,
     in StringTypeOpt ack_system_id,
     out BadAlarmInformationIdSeq
        bad_alarm_information_id_list
 )  raises (UnacknowledgeAlarms,
           error::OperationNotSupported,
           error::ParameterNotSupported,
           error::IlleagalParaException);

/*
Make comment to one or more alarms.
*/
comm::constdef::Signal comment_alarms (
     in AlarmInformationIdSeq alarm_information_id_list,
     in string comment_user_id,
     in StringTypeOpt comment_system_id,
     in string comment_text,
     out BadAlarmInformationIdSeq
          bad_alarm_information_id_list
) raises (CommentAlarms, error::OperationNotSupported,
```

```
            error::ParameterNotSupported,
            error::IlleagalParaException);

/*
Request to clear one or more alarms.
*/
Signal clear_alarms (
        in AlarmInformationIdSeq alarm_information_id_list,
        in string clear_user_id,
        in StringTypeOpt clear_system_id,
        out BadAlarmInformationIdSeq
        bad_alarm_information_id_list
 ) raises (ClearAlarms, error::ParameterNotSupported,
                                error::IlleagalParaException);
/*
This method returns Alarm Informations.                    If flag is
TRUE, all returned Alarm Informations shall be in AlarmInformationSeq that
contains 0 or more Alarm Informations. Output parameter iter shall be useless.
If flag is FALSE, no Alarm Informations shall be in
AlarmInformationSeq . IRPAgent needs to use iter to retrieve them.
*/
AlarmInformationSeq get_alarm_list (
      in StringTypeOpt filter,
) raises (GetAlarmList, error::ParameterNotSupported,
          error::IlleagalParaException);

/*
This method returns the count of Alarm Informations.
*/
void get_alarm_count (
      in StringTypeOpt filter,
      out unsigned long critical_count,
      out unsigned long major_count,
      out unsigned long minor_count,
      out unsigned long warning_count,
      out unsigned long indeterminate_count,
      out unsigned long cleared_count
```

```
        ) raises (GetAlarmCount, error::OperationNotSupported,
                error::ParameterNotSupported,
                error::IlleagalParaException);
    };//AlarmMng
};

#endif
```

A.3.4 安全管理

secumng.idl：

```
#include <error.idl>
#include <comm.idl>
module secumng {
    struct UserInfo{
        string username;
        unsigned long long loginTime;
        unsigned long long logoutTime;
        short loginCount;
    };
    //日志类型
    enum ModuleType{rack,perf,alert,security,log,config,topo,policy};
    //日志类型
    enum LogType{normal,alarm};
    struct LogInfo{
        unsigned long long Id;
        unsigned long long LogTime;
        string ZXUser;
        string HostAddr;
        LogType lt;
        string DevAddr;
        short DevType;
        ModuleType mt;
        string LogDescr;
    };
    typedef sequence<UserInfo> UserInfoList;
    typedef sequence<LogInfo> LogInfoList;
    enum Operation{device,faultmng,mapmng,topomng,admin};
```

```
typedef sequence<Operation> OperationList;
struct UserGroup{
    string groupname;
    OperationList oplist;
};
typedef sequence<UserGroup> UserGroupList;
typedef sequence<string> grouplist;
typedef sequence<string> userlist;
//从根节点到该节点的路径,中间用'.'分隔。（例如：root.shanghai.xuhui）
typedef sequence<string> locpathlist;

typedef sequence<string> domainlist;
interface SecurityMng :comm::CommService {
    //获取指定用户信息
    UserInfo getUserInfo(in string user) raises(error::UserNotExistException);
    //获取所有用户信息
    UserInfoList getAllUserInfo();
    //获取所有日志
    LogInfoList getAllLog();
    //获取指定区间的日志
    LogInfoList getLog(in unsigned long long starttime,in unsigned long long endtime)
        raises(error::IlleagalParaException);
    //创建一个用户组
    boolean createUserGroup(in UserGroup group)
        raises (error::UerGroupAlreadyExistException);
    //修改一指定用户组
    boolean mdUserGroup(in string usergroupname,in OperationList oplist)
        raises (error::UserGroupNotExistException);
    //删除一指定用户组
    boolean delGroup(in string usergroupname)
        raises (error::UserGroupNotExistException);
    //获取所有用户组信息
    UserGroupList getGroup();
    //创建用户
    boolean createUser(in string username,in string password,in grouplist list)
        raises (error::UserAlreadyExistException,error::UserGroupNotExistException);
    //修改一指定用户的密码
```

```
        boolean mdUserPassword(in string username,in string oldpassword,in string newpassword)
          raises (error::UserNotExistException);
        //删除一指定用户
        boolean delUser(in string username) raises (error::UserNotExistException);
        //获取一指定用户组中的所有用户信息
        userlist getUserPerGroup(in string groupname)
          raises (error::UserGroupNotExistException);
        //创建管理域
        boolean createDomain(in string domainname,in locpathlist list)
          raises(error::DomainAlreadyExistException,error::NotAssignLocException);
        //修改一指定管理域
        boolean mdDomain(in string domainname,in locpathlist list)
          raises(error::DomainNotExistException,error::NotAssignLocException);
        //删除一指定管理域
        boolean delDomain(in string domainname)
          raises(error::DomainNotExistException);
        //将一指定管理域分配给一指定用户
        boolean assignDomainToUser(in string username,in domainlist domain)
          raises(error::UserNotExistException,error::DomainNotExistException);
    };
};
```


A.3.5 性能管理

perfmng.idl：

```
#include <error.idl>
#include <comm.idl>
module perfmng{
    /*
    可供采集的性能量定义
    */
    enum AdslPerfPoint{
        /* ATUCSNR 裕度 */
        adslAtucCurrSnrMgn,
        /* ATUC 衰减 */
        adslAtucCurrAtn,
        /* ATUC 状态 */
        adslAtucCurrStatus,
```

```
/* ATUC 输出功率 */
adslAtucCurrOutputPwr,
/* ATUC 可达速率 */
adslAtucCurrAttainableRate,
/* ATURSNR 裕度 */
adslAturCurrSnrMgn,
/* ATUR 衰减 */
adslAturCurrAtn,
/* ATUR 状态 */
adslAturCurrStatus,
/* ATUR 输出功率 */
adslAturCurrOutputPwr,
/* ATUR 可达速率 */
adslAturCurrAttainableRate,
/* 交织延迟 */
adslAtucChanInterleaveDelay,
/* 发送速率 */
adslAtucChanCurrTxRate,
/* 前次连接的发送速率 */
adslAtucChanPrevTxRate,
/* 交织延迟 */
adslAturChanInterleaveDelay,
/* 发送速率 */
adslAturChanCurrTxRate,
/* 前次连接的发送速率 */
adslAturChanPrevTxRate,
/* 帧丢失数 */
adslAtucPerfLofs,
/* 信号丢失数 */
adslAtucPerfLoss,
/* 链路丢失数 */
adslAtucPerfLols,
/* 电源丢失数 */
adslAtucPerfLprs,
/* 误码秒数 */
adslAtucPerfESs,
/* 初始化失败次数 */
```

```
adslAtucPerfInits,
/* 当前 15min 性能统计流逝的时间 */
adslAtucPerfCurr15MinTimeElapsed,
/* 当前 15min 内帧丢失数 */
adslAtucPerfCurr15MinLofs,
/* 当前 15min 内信号丢失数 */
adslAtucPerfCurr15MinLoss,
/* 当前 15min 内链路丢失数 */
adslAtucPerfCurr15MinLols,
/* 当前 15min 内电源丢失数 */
adslAtucPerfCurr15MinLprs,
/* 当前 15min 内误码丢失数 */
adslAtucPerfCurr15MinESs,
/* 当前 15min 内初始化失败次数 */
adslAtucPerfCurr15MinInits,
/* 当前 1 天性能统计流逝的时间 */
adslAtucPerfCurr1DayTimeElapsed,
/* 当前 1 天内帧丢失数 */
adslAtucPerfCurr1DayLofs,
/* 当前 1 天内信号丢失数 */
adslAtucPerfCurr1DayLoss,
/* 当前 1 天内链路丢失数 */
adslAtucPerfCurr1DayLols,
/* 当前 1 天内电源丢失数 */
adslAtucPerfCurr1DayLprs,
/* 当前 1 天内误码丢失数 */
adslAtucPerfCurr1DayESs,
/* 当前 1 天内初始化失败次数 */
adslAtucPerfCurr1DayInits,
/* 前 1 天性能有效的统计时间 */
adslAtucPerfPrev1DayMoniSecs,
/* 前 1 天帧丢失数 */
adslAtucPerfPrev1DayLofs,
/* 前 1 天信号丢失数 */
adslAtucPerfPrev1DayLoss,
/* 前 1 天链路丢失数 */
adslAtucPerfPrev1DayLols,
```

```
/* 前 1 天电源丢失数 */
adslAtucPerfPrev1DayLprs,
/* 前 1 天误码丢失数 */
adslAtucPerfPrev1DayESs,
/* 前 1 天初始化失败次数 */
adslAtucPerfPrev1DayInits,
/* 帧丢失数 */
adslAturPerfLofs,
/* 信号丢失数 */
adslAturPerfLoss,
/* 电源丢失数 */
adslAturPerfLprs,
/* 误码秒数 */
adslAturPerfESs,
/* 当前 15min 性能统计流逝的时间 */
adslAturPerfCurr15MinTimeElapsed,
/* 当前 15min 内帧丢失数 */
adslAturPerfCurr15MinLofs,
/* 当前 15min 内信号丢失数 */
adslAturPerfCurr15MinLoss,
/* 当前 15min 内电源丢失数 */
adslAturPerfCurr15MinLprs,
/* 当前 15min 内误码丢失数 */
adslAturPerfCurr15MinESs,
/* 当前 1 天性能统计流逝的时间 */
adslAturPerfCurr1DayTimeElapsed,
/* 当前 1 天内帧丢失数 */
adslAturPerfCurr1DayLofs,
/* 当前 1 天内信号丢失数 */
adslAturPerfCurr1DayLoss,
/* 当前 1 天内电源丢失数 */
adslAturPerfCurr1DayLprs,
/* 当前 1 天内误码丢失数 */
adslAturPerfCurr1DayESs,
/* 前 1 天性能有效的统计时间 */
adslAturPerfPrev1DayMoniSecs,
/* 前 1 天帧丢失数 */
```

```
adslAturPerfPrev1DayLofs,
/* 前 1 天信号丢失数 */
adslAturPerfPrev1DayLoss,
/* 前 1 天电源丢失数 */
adslAturPerfPrev1DayLprs,
/* 前 1 天误码丢失数 */
adslAturPerfPrev1DayESs,
/* 收到的数据块数 */
adslAtucChanReceivedBlks,
/* 发送的数据块数 */
adslAtucChanTransmittedBlks,
/* 纠正的数据块数 */
adslAtucChanCorrectedBlks,
/* 未纠正的数据块数 */
adslAtucChanUncorrectBlks,
/* 当前 15min 性能统计流逝的时间 */
adslAtucChanPerfCurr15MinTimeElapsed,
/* 当前 15min 收到的数据块数 */
adslAtucChanPerfCurr15MinReceivedBlks,
/* 当前 15min 发送的数据块数 */
adslAtucChanPerfCurr15MinTransmittedBlks,
/* 当前 15min 纠正的数据块数 */
adslAtucChanPerfCurr15MinCorrectedBlks,
/* 当前 15min 未纠正的数据块数 */
adslAtucChanPerfCurr15MinUncorrectBlks,
/* 当前 1 天性能统计流逝的时间 */
adslAtucChanPerfCurr1DayTimeElapsed,
/* 当前 1 天收到的数据块数 */
adslAtucChanPerfCurr1DayReceivedBlks,
/* 当前 1 天发送的数据块数 */
adslAtucChanPerfCurr1DayTransmittedBlks,
/* 当前 1 天纠正的数据块数 */
adslAtucChanPerfCurr1DayCorrectedBlks,
/* 当前 1 天未纠正的数据块数 */
adslAtucChanPerfCurr1DayUncorrectBlks,
/* 前 1 天性能有效的统计时间 */
adslAtucChanPerfPrev1DayMoniSecs,
```

```
/* 前 1 天收到的数据块数 */
adslAtucChanPerfPrev1DayReceivedBlks,
/* 前 1 天发送的数据块数 */
adslAtucChanPerfPrev1DayTransmittedBlks,
/* 前 1 天纠正的数据块数 */
adslAtucChanPerfPrev1DayCorrectedBlks,
/* 前 1 天未纠正的数据块数 */
adslAtucChanPerfPrev1DayUncorrectBlks,
/* 收到的数据块数 */
adslAturChanReceivedBlks,
/* 发送的数据块数 */
adslAturChanTransmittedBlks,
/* 纠正的数据块数 */
adslAturChanCorrectedBlks,
/* 未纠正的数据块数 */
adslAturChanUncorrectBlks,
/* 当前 15min 性能统计流逝的时间 */
adslAturChanPerfCurr15MinTimeElapsed,
/* 当前 15min 收到的数据块数 */
adslAturChanPerfCurr15MinReceivedBlks,
/* 当前 15min 发送的数据块数 */
adslAturChanPerfCurr15MinTransmittedBlks,
/* 当前 15min 纠正的数据块数 */
adslAturChanPerfCurr15MinCorrectedBlks,
/* 当前 15min 未纠正的数据块数 */
adslAturChanPerfCurr15MinUncorrectBlks,
/* 当前 1 天性能统计流逝的时间 */
adslAturChanPerfCurr1DayTimeElapsed,
/* 当前 1 天收到的数据块数 */
adslAturChanPerfCurr1DayReceivedBlks,
/* 当前 1 天发送的数据块数 */
adslAturChanPerfCurr1DayTransmittedBlks,
/* 当前 1 天纠正的数据块数 */
adslAturChanPerfCurr1DayCorrectedBlks,
/* 当前 1 天未纠正的数据块数 */
adslAturChanPerfCurr1DayUncorrectBlks,
/* 前 1 天性能有效的统计时间 */
```

```
    adslAturChanPerfPrev1DayMoniSecs,
    /* 前 1 天收到的数据块数 */
    adslAturChanPerfPrev1DayReceivedBlks,
    /* 前 1 天发送的数据块数 */
    adslAturChanPerfPrev1DayTransmittedBlks,
    /* 前 1 天纠正的数据块数 */
    adslAturChanPerfPrev1DayCorrectedBlks,
    /* 前 1 天未纠正的数据块数 */
    adslAturChanPerfPrev1DayUncorrectBlks,
    ifType,
    ifSpeed,
    ifAdminStatus,
    ifOperStatus,
    ifLastChange,
    ifInOctets,
    ifInErrors,
    ifOutOctets,
    ifOutErrors
};
typedef sequence<unsigned long long> Datalist;
typedef sequence<AdslPerfPoint> AdslPerflist;
enum VdslPerfPoint {
     /* VDSL 当前信噪比裕度 */
    vdslPhysCurrSnrMgn,
     /* VDSL 当前衰减 */
     vdslPhysCurrAtn,
     /* VDSL 当前状态 */
     vdslPhysCurrStatus,
     /* VDSL 当前输出功率 */
       vdslPhysCurrOutputPwr,
       /* VDSL 当前可达速率 */
       vdslPhysCurrAttainableRate,
       /* VDSL 当前速率 */
       vdslPhysCurrLineRate,
       /* 帧丢失秒数 */
       vdslPerfDataLofs,
       /* 信号丢失秒数 */
```

```
vdslPerfDataLoss,
/* 链路丢失秒数 */
vdslPerfDataLols,
/* 电源丢失秒数 */
vdslPerfDataLprs,
/* 误码秒数 */
vdslPerfDataESs,
/* 严重误码秒数 */
vdslPerfDataSESs,
/* 不可达秒数 */
vdslPerfDataUASs,
/* 初始化次数 */
vdslPerfDataInits,
/* 当前 15min 性能统计流逝的时间 */
vdslPerfDataCurr15MinTimeElapsed,
/* 当前 15min 内帧丢失秒数 */
vdslPerfDataCurr15MinLofs,
/* 当前 15min 内信号丢失秒数 */
vdslPerfDataCurr15MinLoss,
/* 当前 15min 内链路丢失秒数 */
vdslPerfDataCurr15MinLols,
/* 当前 15min 内电源丢失秒数 */
vdslPerfDataCurr15MinLprs,
/* 当前 15min 内误码丢失秒数 */
vdslPerfDataCurr15MinESs,
/* 当前 15min 内严重误码秒数 */
vdslPerfDataCurr15MinSESs,
/* 当前 15min 内不可达秒数 */
vdslPerfDataCurr15MinUASs,
/* 当前 15min 内初始化次数 */
vdslPerfDataCurr15MinInits,
/* 当前 1 天性能统计流逝的时间 */
vdslPerfDataCurr1DayTimeElapsed,
/* 当前 1 天内帧丢失秒数 */
vdslPerfDataCurr1DayLofs,
/* 当前 1 天内信号丢失秒数 */
vdslPerfDataCurr1DayLoss,
```

/* 当前 1 天内链路丢失秒数 */
vdslPerfDataCurr1DayLols,
/* 当前 1 天内电源丢失秒数 */
vdslPerfDataCurr1DayLprs,
/* 当前 1 天内误码丢失秒数 */
vdslPerfDataCurr1DayESs,
/* 当前 1 天内严重误码秒数 */
vdslPerfDataCurr1DaySESs,
/* 当前 1 天内不可达秒数 */
vdslPerfDataCurr1DayUASs,
/* 当前 1 天内初始化次数 */
vdslPerfDataCurr1DayInits,
/* 纠正的字节数 */
vdslChanFixedOctets,
/* 未纠正的数据块数 */
vdslChanBadBlks,
/* 当前 15min 性能统计流逝的时间 */
vdslChanCurr15MinTimeElapsed,
/* 当前 15min 内纠正的字节数 */
vdslChanCurr15MinFixedOctets,
/* 当前 15min 内未纠正的数据块数 */
vdslChanCurr15MinBadBlks,
/* 当前 1 天性能统计流逝的时间 */
vdslChanCurr1DayTimeElapsed,
/* 当前 1 天内纠正的字节数 */
vdslChanCurr1DayFixedOctets,
/* 当前 1 天未纠正的数据块数 */
vdslChanCurr1DayBadBlks,
vdslifType,
vdslifSpeed,
vdslifAdminStatus,
vdslifOperStatus,
vdslifLastChange,
vdslifInOctets,
vdslifInErrors,
vdslifOutOctets,
vdslifOutErrors

```
};

typedef sequence<VdslPerfPoint> VdslPerflist;
enum ShdslPerfPoint {
    //端点的量按照端口取
    /* SHDSL 当前信噪比裕度 */
    hdsl2ShdslEndpointCurrSnrMgn,
    /* SHDSL 当前衰减 */
    hdsl2ShdslEndpointCurrAtn,
    /* SHDSL 当前状态 */
    hdsl2ShdslEndpointCurrStatus,
    /* SHDSL 当前中继器个数 */
    hdsl2ShdslStatusNumAvailRepeaters,
    /* SHDSL 当前可达速率 */
    hdsl2ShdslStatusMaxAttainableLineRate,
    /* SHDSL 当前速率 */
    hdsl2ShdslStatusActualLineRate,
    /* SHDSL 当前传输模式 */
    hdsl2ShdslStatusTransmissionModeCurrent,
    /* 误码秒数 */
    hdsl2ShdslEndpointES,
    /* 严重误码秒数 */
    hdsl2ShdslEndpointSES,
    /* CRC 检验异常数 */
    hdsl2ShdslEndpointCRCanomalies,
    /* 同步字丢失秒数 */
    hdsl2ShdslEndpointLOSWS,
    /* 不可达秒数 */
    hdsl2ShdslEndpointUAS,
    /* 当前 15min 性能统计流逝的时间 */
    hdsl2ShdslEndpointCurr15MinTimeElapsed,
    /* 当前 15min 内误码秒数 */
    hdsl2ShdslEndpointCurr15MinES,
    /* 当前 15min 内严重误码秒数 */
    hdsl2ShdslEndpointCurr15MinSES,
    /* 当前 15min 内 CRC 检验异常数 */
    hdsl2ShdslEndpointCurr15MinCRCanomalies,
```

```
    /* 当前 15min 内同步字丢失秒数 */
    hdsl2ShdslEndpointCurr15MinLOSWS,
    /* 当前 15min 内不可达秒数 */
    hdsl2ShdslEndpointCurr15MinUAS,
    /* 当前 1 天性能统计流逝的时间 */
    hdsl2ShdslEndpointCurr1DayTimeElapsed,
    /* 当前 1 天内误码秒数 */
    hdsl2ShdslEndpointCurr1DayES,
   /* 当前 1 天内严重误码秒数 */
   hdsl2ShdslEndpointCurr1DaySES,
   /* 当前 1 天内 CRC 检验异常数 */
   hdsl2ShdslEndpointCurr1DayCRCanomalies,
   /* 当前 1 天内同步字丢失秒数 */
   hdsl2ShdslEndpointCurr1DayLOSWS,
   /* 当前 1 天内不可达秒数 */
   hdsl2ShdslEndpointCurr1DayUAS,
   //对下面这几个量来说，只能按照端口取
   hdsl2ShdslifType,
   hdsl2ShdslifSpeed,
   hdsl2ShdslifAdminStatus,
   hdsl2ShdslifOperStatus,
   hdsl2ShdslifLastChange,
   hdsl2ShdslifInOctets,
   hdsl2ShdslifInErrors,
   hdsl2ShdslifOutOctets,
   hdsl2ShdslifOutErrors
};
struct ShdslEndPointIndex{
    short cardno;
    short portno;
    /**
    * 端点，其中 STUC 和 STUR 是肯定有的，Repeater1~8 有几个需要根据<br>
    *   hdsl2ShdslStatusNumAvailRepeaters 判断，取值范围：<br>
    * xtuC = 1,          // STUC <br>
    * xtuR = 2,          // STUR <br>
    * xru1 = 3,          // Repeater1<br>
    * xru2 = 4,          // Repeater2<br>
```

```
        * xru3 = 5,          // Repeater3<br>
        * xru4 = 6,          // Repeater4 <br>
        * xru5 = 7,          // Repeater5<br>
        * xru6 = 8,          // Repeater6<br>
        * xru7 = 9,          // Repeater7<br>
        * xru8 = 10          // Repeater8<br>
        * */
        short                hdsl2ShdslInvIndex;          // index

        /**
        *端点侧，取值范围：<br>
        * networkSide = 1,          // on the Network side <br>
        * customerSide = 2    // on the Customer side <br>
        * */
        //对 xtuc 来说，只有 customerSide，对 xtur 来说，只有 networkSide
        short                hdsl2ShdslEndpointSide;          // index

        /**
        *端点线对，取值范围：<br>
        * wirePair1 = 1,          // the first pair <br>
        * wirePair2 = 2           // the optional second pair <br>
        * */
        short                hdsl2ShdslEndpointWirePair;          // index
};
typedef sequence<ShdslPerfPoint> ShdslEndPointPerflist;
interface PerfMng{
    ///获取指定端口上指定采集点的性能量取值
    Datalist collectPerf(in string neipaddr,in short cardno,in short portno,
      in AdslPerflist collectpoint)
            raises(error::NeNotExistException,error::IlleagalParaException );
    Datalist collectVdslCurrPerf(in string neipaddr,in short cardno,in short portno,
      in VdslPerflist collectpoint)
            raises(error::NeNotExistException,error::IlleagalParaException );
    Datalist collectShdslCurrPerf(in string neipaddr,in ShdslEndPointIndex endpointindex, in
          ShdslEndPointPerflist collectpoint)
            raises(error::NeNotExistException,error::IlleagalParaException );
};
```

```
};
```

A.3.6 异常定义

error.idl：

```
module error{
    /*
    Exception thrown when an unsupported optional parameter
    is passed with information.
    The parameter shall be the actual unsupported parameter name.
    */
    exception ParameterNotSupported { string parameter; };

    /*
    Exception thrown when a valid but unsupported parameter value is passed.
   The parameter shall be the actual parameter name.
   */
   exception ValueNotSupported { string parameter; };

   /*
   Exception thrown when an unsupported optional method is called.
   */
   exception OperationNotSupported {};
   exception UserNotExistException{
       string message;
    };
    exception NeNotExistException{
        string message;
    };
    exception AlreadyRegisteredException{
        string message;
    };
    exception NotRegisteredException{
        string message;
     };
     /*
      Exception thrown when an invalid parameter value is passed.
      The parameter shall be the actual parameter name.
```

```
*/
exception IlleagalParaException{
    string message;
};
exception UerGroupAlreadyExistException{
    string message;
};
exception UserGroupNotExistException{
    string message;
};
exception UserAlreadyExistException{
    string message;
};
exception DuplicateFilterNoException{
    string message;
};
exception FilterNotExistException{
    string message;
};
exception EmsInternalException{
    string message;
};
exception DomainAlreadyExistException{
    string message;
};
exception NotAssignLocException{
    string message;
};
exception DomainNotExistException{
    string message;
};
exception DomainNotAssignedException{
    string message;
};
exception TasknoNotExistException{
    string message;
};
```

```
        exception NoRightException{
            string message;
        };
    };
```

A.3.7 通用常量定义

commconst.idl

```
#ifndef _CommConstDef_idl_
#define _CommConstDef_idl_

        module comm{
            module constdef{
                /*common const will be moved to managedgeneric*/
                /*
                Request to acknowledge one or more alarms.
                */

                enum Signal {OK, Failure, PartialFailure};

                /*
                StringTypeOpt is a type carrying an optional parameter.
                If the boolean is TRUE, then the value is present.
                Otherwise the value is absent.
                */
                union StringTypeOpt switch (boolean)
                {
                    case TRUE: string value;
                };

                /*
                ShortTypeOpt is a type carrying an optional parameter.
                If the boolean is TRUE, then the value is present.
                Otherwise the value is absent.
                */
                union ShortTypeOpt switch (boolean)
                {
                    case TRUE: short value;
```

```
            };
        };
    };
#endif
```

A.3.8 实现建议

为了让 NMS 访问服务的可扩展性，一致性和易操作性以及 EMS 服务实现的多样性，不同设备，不同业务类型服务的易配置和易添加性，采用了一些技术。

扩展接口：是一种运用在框架中的常用技术。该技术可以使得接口的功能变得可扩展而不会影响到已有的使用者。为了达到这个目的，扩展接口技术提供了一个高层的、一致的抽象服务接口。在附录提供的公共管理（comm.idl）中就定义了这样一个退化了的扩展接口：CommService。所有的服务都要继承这个扩展接口，使得服务的访问能够一致。

采用配置文件的方式方便对服务的组织，服务的配置发布不再采用硬编码的方式，而是采用动态加载和控制反转（Inversion of Control）技术使得服务的表示和实现分离，从而可以很方便地实现服务的多样性，也方便实现服务的热部署。

```
<servicereg>
    <specialservice servicetype="conf" >
        <businessimp type ="adsl"
        imp="com.device.dsl.common.corba.conf.ConfigMngImpl">
            <neproc setter="setProcI">
                <item type="DSLSpec1"
                    procimp="com.device.dsl.dslspecific.corba.AdslSpecific.ConfDataGetImpl"/>
            </neproc>
        </businessimp>
    </specialservice>
</servicereg>
```

提供服务管理入口服务 ServiceMng 来管理服务的一致性获取。

中华人民共和国
通信行业标准
数字用户线路（DSL）网络管理接口技术要求
YD/T 1665-2007

*

人民邮电出版社出版发行
北京市崇文区夕照寺街14号A座
邮政编码：100061
北京新瑞铭印刷有限公司印刷

*

开本：880×1230　1/16　　2007年12月第1版
印张：8.25　　2007年12月北京第1次印刷
字数：254千字

ISBN 978-7-115-1540/07－203

定价：60元

本书如有印装质量问题，请与本社联系　电话：(010)67114922