

ICS 33.040

M 15

YD

中华人民共和国通信行业标准

YD/T 2149.5-2011

光传送网（OTN）网络管理技术要求 第5部分：基于IDL/IIOP技术的 EMS-NMS 接口信息模型

Technical requirement for optical transport network management
—Part 5: EMS-NMS interface information model based on IDL/IIOP

2011-12-20 发布

2012-02-01 实施

中华人民共和国工业和信息化部 发布

目 次

| | |
|----------------------|-----|
| 前 言 | II |
| 1 范围 | 1 |
| 2 规范性引用文件 | 1 |
| 3 术语、定义和缩略语 | 1 |
| 3.1 术语和定义 | 1 |
| 3.2 缩略语 | 1 |
| 4 接口信息描述模板 | 2 |
| 4.1 数据定义描述模板 | 2 |
| 4.2 接口定义描述模板 | 2 |
| 5 接口规范 | 2 |
| 5.1 配置管理模块 | 2 |
| 5.2 故障管理模块 | 68 |
| 5.3 性能管理模块 | 71 |
| 5.4 通用管理模块 | 85 |
| 5.5 公共管理模块 | 92 |
| 附录A（规范性附录） 模块与IDL的映射 | 105 |

前 言

《光传送网（OTN）网络管理技术要求》分为6个部分：

- 光传送网（OTN）网络管理技术要求 第1部分：基本原则；
- 光传送网（OTN）网络管理技术要求 第2部分：NMS系统功能；
- 光传送网（OTN）网络管理技术要求 第3部分：EMS-NMS接口功能；
- 光传送网（OTN）网络管理技术要求 第4部分：EMS-NMS接口通用信息模型；
- 光传送网（OTN）网络管理技术要求 第5部分：基于IDL/IOP技术的EMS-NMS接口信息模型；
- 光传送网（OTN）网络管理技术要求 第6部分：基于XML技术的EMS-NMS接口信息模型。

本部分是第5部分。

本部分按照GB/T 1.1-2009给出的规则起草。

本部分由中国通信标准化协会提出并归口。

本部分起草单位：北京邮电大学、北京市天元网络技术股份有限公司、中兴通讯股份有限公司、华为技术有限公司、武汉邮电科学研究院、上海贝尔股份有限公司。

本部分主要起草人：陈兴渝、熊 翱、年庆飞、鹿虹丽、元 峰、强国武、赵纯利、扶文忠、张丽雅、蒙向阳、张 励。

光传送网（OTN）网络管理技术要求

第5部分：基于IDL/IIOP技术的EMS-NMS接口信息模型

1 范围

本部分规定了光传送网络（OTN）网络管理体系EMS-NMS之间基于IDL/IIOP技术的接口规范。
本部分适用于OTN网络管理系统的多厂商互通的实现。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅所注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

| | |
|------------------|----------------------------------|
| YD/T 2149.2-2010 | OTN网络管理技术要求 第2部分：NMS系统功能 |
| YD/T 2149.3-2010 | OTN网络管理技术要求 第3部分：EMS-NMS接口功能 |
| YD/T 2149.4-2010 | OTN网络管理技术要求 第4部分：EMS-NMS接口通用信息模型 |
| TMF814（2003） | 多技术网络管理解决方案 NML-EML接口 版本3.0 |

3 术语、定义和缩略语

3.1 术语和定义

YD/T 2149.2-2010和YD/T 2149.3-2010的3.1中定义的术语适用于本文件。

3.2 缩略语

下列缩略语适用于本文件。

| | | |
|------|-----------------------------------|---------------|
| ASAP | Alarm Severity Assignment Profile | 告警级别分配模板 |
| CTP | Connection Termination Point | 连接终端点 |
| EMS | Element Management System | 网元管理系统 |
| FTP | Floating Termination Point | 浮动终端点 |
| IDL | Interface Description Language | 接口描述语言 |
| IIOP | Internet Inter-ORB Protocol | 互联网内部对象请求代理协议 |
| ME | Managed Element | 网元 |
| MFD | Matrix Flow Domain | 矩阵流域 |
| NMS | Network Management System | 网络管理系统 |
| OADM | Optical Add-Drop Multiplexer | 光分插复用器 |
| OLA | Optical Line Amplifier | 光线路放大器 |
| OPM | Optical Performance Monitor | 光性能监测 |
| OTM | Optical Termination Multiplexer | 光终端复用器 |
| OTU | Optical Tranponder Unit | 光转换器单元 |
| PTP | Physical Termination Point | 物理终端点 |
| SNMS | SubNetwork Management System | 子网管理系统 |
| TP | Termination Point | 终端点 |

4 接口信息描述模板

4.1 数据定义描述模板

| | |
|--------------|---------|
| 定义 | |
| 数据结构的 IDL 定义 | |
| 说明 | |
| 对象说明 | 数据对象的说明 |
| 属性名 | 属性说明 |
| 属性名称 | 属性含义说明 |

4.2 接口定义描述模板

| | |
|--------------|-------------|
| 定义 | |
| 接口操作的 IDL 定义 | |
| 说明 | |
| 功能描述 | 接口功能描述说明 |
| 输入参数 | 输入参数含义说明 |
| 输入/输出参数 | 输入/输出参数含义说明 |
| 输出参数 | 输出参数含义说明 |
| 操作异常 | 异常说明 |

本部分中各模块与IDL文件间的相互关系见附录A。

5 接口规范

5.1 配置管理模块

5.1.1 EMS 管理模块 (module emsMgr)

5.1.1.1 EMS (EMS_T)

| | |
|---|-----------|
| 定义 | |
| <pre> struct EMS_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; string emsVersion; string type; globaldefs::NVSList_T additionalInfo; }; </pre> | |
| 说明 | |
| 对象说明 | EMS 信息 |
| 属性名 | 属性说明 |
| name | EMS 的名称 |
| userLabel | EMS 的用户标签 |
| nativeEMSName | EMS 的本地名称 |
| owner | EMS 的所有者 |
| emsVersion | EMS 的软件版本 |
| type | 厂家自定义类型 |

| | |
|----------------|---|
| additionalInfo | 附加信息。包括： <ul style="list-style-type: none"> — 子类型，包括“EMS”、“SNMS”； — EMS 厂商名称； — EMS 的 IP 地址及与 NMS 相连的通信端口号； — EMS 接口版本； — EMS 所在的地理位置（精确到机房） |
|----------------|---|

5.1.1.2 EMSMgr_I 接口（从 common::Common_I 继承）

5.1.1.2.1 查询 EMS 信息（getEMS）

| | |
|---|----------------------|
| 定义 | |
| <pre>void getEMS (out EMS_T emsInfo) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | NMS 向 EMS 查询 EMS 信息 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | emsInfo:: EMS 信息。 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.1.2.2 查询所有顶层子网（getAllTopLevelSubnetworks）

| | |
|--|---|
| 定义 | |
| <pre>void getAllTopLevelSubnetworks (in unsigned long how_many, out multiLayerSubnetwork::SubnetworkList_T sList, out multiLayerSubnetwork::SubnetworkIterator_I sIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询 EMS 内的所有顶层子网信息。顶层子网指包含物理层且位于子网包含关系的最顶层的子网 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | sList: 顶层子网列表。 sIt: 迭代查询顶层子网接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_TOO_MANY_OPEN_ITERATORS EXCPT_NE_COMM_LOSS |

5.1.1.2.3 查询所有顶层子网名称（getAllTopLevelSubnetworkNames）

| | |
|--|--|
| 定义 | |
| <pre>void getAllTopLevelSubnetworkNames (in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |

| 说明 | |
|---------|---|
| 功能描述 | 查询 EMS 内的所有顶层子网名称 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 顶层子网名称列表。 nameIt: 迭代查询子网名称接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_TOO_MANY_OPEN_ITERATORS EXCPT_NE_COMM_LOSS |

5.1.1.2.4 查询所有顶层拓扑连接 (getAllTopLevelTopologicalLinks)

| 定义 | |
|--|--|
| <pre>void getAllTopLevelTopologicalLinks (in unsigned long how_many, out topologicalLink::TopologicalLinkList_T topoList, out topologicalLink::TopologicalLinkIterator_I topoIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询所有顶层拓扑连接。这里的顶层拓扑连接指 EMS 管理的子网间的拓扑连接 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无。 |
| 输出参数 | topoList: 拓扑连接列表。 topoIt: 迭代查询拓扑连接接口 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.1.2.5 查询所有顶层拓扑连接名称 (getAllTopLevelTopologicalLinkNames)

| 定义 | |
|--|--|
| <pre>void getAllTopLevelTopologicalLinkNames (in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询所有顶层拓扑连接的名称 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 拓扑连接名称列表。 nameIt: 迭代查询名称接口 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.1.2.6 查询指定的顶层拓扑连接 (getTopLevelTopologicalLink)

| 定义 | |
|---|--|
| <pre>void getTopLevelTopologicalLink (in globaldefs::NamingAttributes_T topoLinkName, out topologicalLink::TopologicalLink_T topoLink) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 指定拓扑连接名称查询拓扑连接信息 |
| 输入参数 | topoLinkName: 拓扑链接名称 |
| 输入/输出参数 | 无 |
| 输出参数 | topoLink: 拓扑连接 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND |

5.1.1.2.7 查询所有当前告警 (getAllEMSAndMEActiveAlarms)

| 定义 | |
|--|--|
| <pre>void getAllEMSAndMEActiveAlarms (in notifications::ProbableCauseList_T excludeProbCauseList, in notifications::PerceivedSeverityList_T excludeSeverityList, in unsigned long how_many, out notifications::EventList_T eventList, out notifications::EventIterator_I eventIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询所有当前告警 (包括越门限告警), 包括所有网元以及 EMS 本身产生的告警。查询条件包括告警原因和告警级别 |
| 输入参数 | excludeProbCauseList: 表示被排除的告警级别列表。如果列表是空, 表示不排除。 excludeSeverityList: 表示被排除的告警原因列表。如果列表是空, 表示不排除。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | eventList: 符合条件的当前告警列表。 eventIt: 获得剩余查询数据的事件迭代器操作接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.1.2.8 查询 EMS 所有当前告警 (getAllEMSSystemActiveAlarms)

| 定义 | |
|---|--|
| <pre>void getAllEMSSystemActiveAlarms (in notifications::PerceivedSeverityList_T excludeSeverityList, in unsigned long how_many, out notifications::EventList_T eventList, out notifications::EventIterator_I eventIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |

| 说明 | |
|---------|--|
| 功能描述 | 查询所有 EMS 本身产生的当前告警 |
| 输入参数 | excludeSeverityList: 表示被排除的告警原因列表, 如果列表是空, 表示不排除。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | eventList: 符合条件的当前告警列表。 eventIt: 获得剩余查询数据的事件迭代器操作接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.1.2.9 创建告警级别分配模板 (createASAP) (可选)

| 定义 | |
|--|--|
| <pre>void createASAP (in aSAP::ASAPCreateModifyData_T newASAPCreateData, out aSAP::ASAP_T newASAP, out globaldefs::NVSList_T additionalInfo) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 创建告警级别分配模板 |
| 输入参数 | newASAPCreateData: ASAP 创建数据 |
| 输入/输出参数 | 无 |
| 输出参数 | newASAP: 新创建的 ASAP。 additionalInfo: 附加信息 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INVALID_INPUT EXCPT_INTERNAL_ERROR EXCPT_UNABLE_TO_COMPLY |

5.1.1.2.10 删除告警级别分配模板 (deleteASAP) (可选)

| 定义 | |
|---|---|
| <pre>void deleteASAP (in globaldefs::NamingAttributes_T aSAPName, inout globaldefs::NVSList_T additionalInfo) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 删除告警级别分配模板 |
| 输入参数 | aSAPName: 指定的要删除的告警级别分配模板名称 |
| 输入/输出参数 | additionalInfo: 附加信息 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_OBJECT_IN_USE EXCPT_UNABLE_TO_COMPLY |

5.1.1.2.11 分配告警级别分配模板 (assignASAP) (可选)

| 定义 | |
|---|--|
| <pre>void assignASAP (in globaldefs::NamingAttributes_T aSAPName, in globaldefs::NamingAttributes_T resourceName, in transmissionParameters::LayerRate_T layerRate, inout globaldefs::NVSList_T additionalInfo) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 分配告警级别分配模板 |
| 输入参数 | aSAPName: 告警级别分配模板名称。 resourceName: 将要分配 ASAP 模板的资源名称。 layerRate: 分配 ASAP 模板的资源的速率 |
| 输入/输出参数 | additionalInfo: 附加信息 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY |

5.1.1.2.12 去分配告警级别分配模板 (deassignASAP) (可选)

| 定义 | |
|---|--|
| <pre>void deassignASAP (in globaldefs::NamingAttributes_T resourceName, in transmissionParameters::LayerRate_T layerRate, inout globaldefs::NVSList_T additionalInfo) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 去分配告警级别分配模板 |
| 输入参数 | resourceName: 将要去分配 ASAP 模板的资源名称。 layerRate: 去分配 ASAP 模板的资源的速率 |
| 输入/输出参数 | additionalInfo: 附加信息 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY |

5.1.1.2.13 修改告警级别分配模板 (modifyASAP) (可选)

| 定义 | |
|---|--|
| <pre>void modifyASAP (in globaldefs::NamingAttributes_T aSAPName, in aSAP::ASAPCreateModifyData_T aSAPModifyData, out aSAP::ASAP_T newASAP, out globaldefs::NVSList_T additionalInfo) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 修改告警级别分配模板 |
| 输入参数 | aSAPName: 告警级别分配模板名称。 aSAPModifyData: 告警级别分配模板修改数据 |
| 输入/输出参数 | 无 |
| 输出参数 | newASAP: 修改后的告警级别分配模板。 additionalInfo: 附加信息 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY |

5.1.1.2.14 查询所有告警级别分配模板 (getAllASAPs) (可选)

| 定义 | |
|---|--|
| <pre>void getAllASAPs (in unsigned long how_many, out aSAP::ASAPList_T aSAPList, out aSAP::ASAPIterator_I asapIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询所有告警级别分配模板 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | aSAPList: 告警级别分配模板列表。 asapIt: 迭代查询告警级别分配模板接口 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.1.2.15 查询所有告警级别分配模板名称 (getAllASAPNames) (可选)

| 定义 | |
|---|--|
| <pre>void getAllASAPNames (in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException);</pre> | |

| 说明 | |
|---------|--|
| 功能描述 | 查询所有告警级别分配模板名称 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 迭代查询名称接口 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.1.2.16 查询指定的告警级别分配模板 (getASAP) (可选)

| 定义 | |
|--|--|
| <pre>void getASAP (in globaldefs::NamingAttributes_T aSAPName, out aSAP::ASAP_T anASAP) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 指定告警级别分配模板名称, 查询告警级别分配模板 |
| 输入参数 | aSAPName: 告警级别分配模板名称 |
| 输入/输出参数 | 无 |
| 输出参数 | anASAP: 告警级别分配模板 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND |

5.1.1.2.17 查询与指定资源匹配的告警级别分配模板 (getASAPbyResource) (可选)

| 定义 | |
|--|---|
| <pre>void getASAPbyResource (in globaldefs::NamingAttributes_T resourceName, in transmissionParameters::LayerRateList_T layerRateList, in unsigned long how_many, out aSAP::ASAPList_T aSAPList, out aSAP::ASAPIterator_I asapIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询与指定资源匹配的告警级别分配模板 |
| 输入参数 | resourceName: 资源名称。 layerRateList: 层速率列表。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | aSAPList: 告警级别分配模板列表。 asapIt: 查询告警级别分配接口 |

| | |
|------|---|
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_TOO_MANY_OPEN_ITERATORS |
|------|---|

5.1.1.2.18 查询告警级别分配模板相关资源的名称 (getASAPAssociatedResourceNames) (可选)

| | |
|--|--|
| 定义 | |
| <pre>void getASAPAssociatedResourceNames (in globaldefs::NamingAttributes_T objectName, out globaldefs::NamingAttributesList_T resourceNameList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 指定对象名称，查询该对象的告警级别分配模板相关资源的名称 |
| 输入参数 | objectName: 对象名称 |
| 输入/输出参数 | 无 |
| 输出参数 | resourceNameList: 查询获得的资源名称列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS |

5.1.2 拓扑连接模块 (module topologicalLink)

5.1.2.1 拓扑连接 (TopologicalLink_T)

| | |
|---|-----------|
| 定义 | |
| <pre>struct TopologicalLink_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; globaldefs::ConnectionDirection_T direction; transmissionParameters::LayerRate_T rate; globaldefs::NamingAttributes_T aEndTP; globaldefs::NamingAttributes_T zEndTP; globaldefs::NVList_T additionalInfo; };</pre> | |
| 说明 | |
| 对象说明 | 拓扑连接信息 |
| 属性名 | 属性说明 |
| name | 拓扑连接名称 |
| userLabel | 拓扑连接的用户标签 |

| | |
|----------------|--|
| nativeEMSName | 拓扑连接的本地名称 |
| owner | 拓扑连接的所有者 |
| direction | 拓扑连接的方向 |
| rate | 拓扑连接的速率 |
| aEndTP | 拓扑连接的 A 点名称 |
| zEndTP | 拓扑连接的 Z 点名称 |
| additionalInfo | 附加信息。包括： — 子类型，包括“网元内拓扑连接”、“网元间拓扑连接”、“跨 EMS 拓扑连接”（可选） |

5.1.2.2 拓扑连接列表 (TopologicalLinkList_T)

| | |
|--|--------|
| 定义 | |
| typedef sequence<TopologicalLink_T> TopologicalLinkList_T; | |
| 说明 | |
| 对象说明 | 拓扑连接列表 |

5.1.2.3 TopologicalLinkIterator_I 接口

5.1.2.3.1 从迭代器中查询数据 (next_n)

| | |
|---|---|
| 定义 | |
| <pre>boolean next_n (in unsigned long how_many, out TopologicalLinkList_T topoLinkList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时，系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | topoLinkList: 首次查询返回的拓扑连接列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.2.3.2 查询迭代器记录条数 (getLength)

| | |
|--|---|
| 定义 | |
| <pre>unsigned long getLength () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数（注意这里指的是迭代器中数据记录的总条数，该值在迭代器生命周期内是不变的） |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.2.3.3 销毁迭代器对象 (destroy)

| | |
|--|----------------------|
| 定义 | |
| <pre>void destroy () raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.3 网元模块 (module managedElement)

5.1.3.1 通信状态 (CommunicationState_T)

| | |
|--|-------------------------------|
| 定义 | |
| <pre>enum CommunicationState_T { CS_AVAILABLE, CS_UNAVAILABLE };</pre> | |
| 说明 | |
| 对象说明 | 表示系统或模块之间 (如 EMS 与网元之间) 的通信状态 |
| 类型取值 | 取值说明 |
| CS_AVAILABLE | 通讯正常 |
| CS_UNAVAILABLE | 通讯失效 |

5.1.3.2 网元 (ManagedElement_T)

| | |
|---|------|
| 定义 | |
| <pre>struct ManagedElement_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; string location; string version; string productName; CommunicationState_T communicationState; boolean emsInSyncState; transmissionParameters::LayerRateList_T supportedRates; globaldefs::NVList_T additionalInfo; };</pre> | |
| 说明 | |
| 对象说明 | 网元信息 |
| 属性名 | 属性说明 |
| name | 网元名称 |

| | |
|--------------------|---|
| userLabel | 网元的用户标签 |
| nativeEMSName | 网元的本地名称 |
| owner | 网元的所有者 |
| location | 网元的地理位置 |
| version | 网元的软件版本 |
| productName | 网元型号 |
| communicationState | EMS 与网元的通信状态 |
| emsInSyncState | 表示 EMS 是否能够保持 EMS 数据与网元数据同步，并且能够发送所有适当的的通知。当 EMS 将此属性设置为假时，表示 EMS 需要与网元进行数据同步并且不能产生适当的通知（如对象创建、删除、属性改变通知）；当 EMS 将此属性设置回到真表示数据重新同步结束同时通知可以在适当的时候产生 |
| supportedRates | 网元支持的交叉连接速率 |
| additionalInfo | 附加信息。包括： <ul style="list-style-type: none"> — 子类型，包括：OADM、OTM、OLA、虚拟网元等；（可选） — 网元的硬件版本；（可选） — 网元供应商名称 |

5.1.3.3 网元列表 (ManagedElementList_T)

| | |
|--|------|
| 定义 | |
| typedef sequence<ManagedElement_T> ManagedElementList_T; | |
| 说明 | |
| 对象说明 | 网元列表 |

5.1.3.4 ManagedElementIterator_I 接口

5.1.3.4.1 从迭代器中查询数据 (next_n)

| | |
|--|---|
| 定义 | |
| <pre>boolean next_n (in unsigned long how_many, out ManagedElementList_T meList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时，系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | meList: 首次查询返回的网元列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.3.4.2 查询迭代器记录条数 (getLength)

| | |
|--|---|
| 定义 | |
| <pre>unsigned long getLength () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数（注意这里指的是迭代器中数据记录的总条数，该值在迭代器生命周期内是不变的） |

| | |
|---------|------------------------------|
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.3.4.3 销毁迭代器对象 (destroy)

| | |
|---|----------------------|
| 定义 | |
| <pre>void destroy (raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.4 网元管理模块 (module managedElementManager)

5.1.4.1 ManagedElementMgr_I 接口 (从 common::Common_I 继承)

5.1.4.1.1 查询 EMS 下所有网元 (getAllManagedElements)

| | |
|--|---|
| 定义 | |
| <pre>void getAllManagedElements (in unsigned long how_many, out managedElement::ManagedElementList_T meList, out managedElement::ManagedElementIterator_I meIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询 EMS 下所有网元 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | meList: 网元列表。 meIt: 网元迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.4.1.2 查询 EMS 下所有网元名称 (getAllManagedElementNames)

| | |
|--|----------------|
| 定义 | |
| <pre>void getAllManagedElementNames (in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询 EMS 下所有网元名称 |

| | |
|---------|---|
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 名称迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.4.1.3 查询指定的网元 (getManagedElement)

| | |
|---|---|
| 定义 | |
| <pre>void getManagedElement (in globaldefs::NamingAttributes_T managedElementName, out managedElement::ManagedElement_T me) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 指定名称查询网元 |
| 输入参数 | managedElementName: 网元名称 |
| 输入/输出参数 | 无 |
| 输出参数 | me: 网元信息 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.4.1.4 查询网元下所有的 PTP 信息 (getAllPTPs)

| | |
|--|---|
| 定义 | |
| <pre>void getAllPTPs (in globaldefs::NamingAttributes_T managedElementName, in transmissionParameters::LayerRateList_T tpLayerRateList, in transmissionParameters::LayerRateList_T connectionLayerRateList, in unsigned long how_many, out terminationPoint::TerminationPointList_T tpList, out terminationPoint::TerminationPointIterator_I tpIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询网元下所有的 PTP |
| 输入参数 | managedElementName: 网元名称。 tpLayerRateList: 要求满足的 PTP 速率等级列表, 查询结果的 PTP 的层速率至少要符合列表中的一项, 如果是空, 则没有限制。 connectionLayerRateList: 要求满足的连接层速率等级列表, 查询出的终端点的连接层速率至少要符合列表中的一项, 如果是空, 则没有限制。 how_many: 首次迭代查询的数目 |
| 输入/输出参数 | 无 |
| 输出参数 | tpList: 查询获得的终端点列表。 tpIt: 迭代查询终端点接口 |

| | |
|------|--|
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |
|------|--|

5.1.4.1.5 查询网元下所有的 PTP 名称 (getAllPTPNames)

| | |
|---|---|
| 定义 | |
| <pre>void getAllPTPNames (in globaldefs::NamingAttributes_T managedElementName, in transmissionParameters::LayerRateList_T tpLayerRateList, in transmissionParameters::LayerRateList_T connectionLayerRateList, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询网元下所有的 PTP 名称 |
| 输入参数 | managedElementName: 网元名称。 tpLayerRateList: 要求满足的 PTP 速率等级列表, 查询返回名称的 PTP 的层速率至少要符合列表中的一 项, 如果是空, 则没有限制。 connectionLayerRateList: 要求满足的连接层速率等级列表, 查询出名称的终端点的连接层速率至少 要符合列表中的一 项, 如果是空, 则没有限制。 how_many: 首次迭代查询的数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 名称迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.4.1.6 查询指定的 TP (getTP)

| | |
|--|---------------|
| 定义 | |
| <pre>void getTP (in globaldefs::NamingAttributes_T tpName, out terminationPoint::TerminationPoint_T tp) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 指定名称查询 TP 信息 |
| 输入参数 | tpName: TP 名称 |
| 输入/输出参数 | 无 |
| 输出参数 | tp: TP 信息 |

| | |
|------|---|
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |
|------|---|

5.1.4.1.7 查询所属 TP 信息 (getContainingTPs)

| | |
|---|--|
| 定义 | |
| <pre>void getContainingTPs (in globaldefs::NamingAttributes_T tpName, out terminationPoint::TerminationPointList_T tpList) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询所属 TP 信息 |
| 输入参数 | tpName: TP 名称 |
| 输入/输出参数 | 无 |
| 输出参数 | tpList: TP 列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.4.1.8 查询所属 TP 名称 (getContainingTPNames)

| | |
|---|--|
| 定义 | |
| <pre>void getContainingTPNames (in globaldefs::NamingAttributes_T tpName, out globaldefs::NamingAttributesList_T tpNameList) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询所属 TP 名称 |
| 输入参数 | tpName: TP 名称 |
| 输入/输出参数 | 无 |
| 输出参数 | tpNameList: TP 名称列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.4.1.9 查询指定网元下所有的当前告警 (getAllActiveAlarms)

| | |
|---|--|
| 定义 | |
| <pre>void getAllActiveAlarms (in globaldefs::NamingAttributes_T meName, in notifications::ProbableCauseList_T excludeProbCauseList, in notifications::PerceivedSeverityList_T excludeSeverityList,</pre> | |

| | |
|--|---|
| <pre> in unsigned long how_many, out notifications::EventList_T eventList, out notifications::EventIterator_I eventIt) raises (globaldefs::ProcessingFailureException); </pre> | |
| 说明 | |
| 功能描述 | 查询指定网元下所有的当前告警 |
| 输入参数 | meName: 网元名称。 excludeProbCauseList: 表示被排除的告警级别列表, 如果列表是空, 表示不排除。 excludeSeverityList: 表示被排除的告警原因列表, 如果列表是空, 表示不排除。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | eventList: 符合条件的当前告警列表。 eventIt: 获得剩余查询数据的事件迭代器操作接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.4.1.10 查询指定网元下所有的交叉连接 (getAllCrossConnections)

| | |
|---|--|
| 定义 | |
| <pre> void getAllCrossConnections (in globaldefs::NamingAttributes_T managedElementName, in transmissionParameters::LayerRateList_T connectionRateList, in unsigned long how_many, out subnetworkConnection::CrossConnectList_T ccList, out subnetworkConnection::CCIterator_I ccIt) raises (globaldefs::ProcessingFailureException); </pre> | |
| 说明 | |
| 功能描述 | 查询网元下所有的交叉连接 |
| 输入参数 | meName: 网元名称。 connectionRateList: 交叉连接支持的速率列表。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | ccList: 交叉连接列表。 ccIt: 交叉连接迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.4.1.11 查询网元所属的子网名称 (getContainingSubnetworkNames) (可选)

| | |
|---|---|
| 定义 | |
| <pre>void getContainingSubnetworkNames (in globaldefs::NamingAttributes_T managedElementName, out globaldefs::NamingAttributesList_T subnetNames) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询 ME 所属的子网名称 |
| 输入参数 | managedElementName: 网元名称 |
| 输入/输出参数 | 无 |
| 输出参数 | subnetNames: 子网名称列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_COMM_LOSS |

5.1.5 设备模块 (module equipment)

5.1.5.1 设备类型 (EquipmentObjectType_T)

| | |
|--|-------------|
| 定义 | |
| <pre>typedef string EquipmentObjectType_T;</pre> | |
| 说明 | |
| 对象说明 | 设备类型, 由厂家定义 |

5.1.5.2 设备类型列表 (EquipmentObjectTypeList_T)

| | |
|---|--------|
| 定义 | |
| <pre>typedef sequence<EquipmentObjectType_T> EquipmentObjectTypeList_T;</pre> | |
| 说明 | |
| 对象说明 | 设备类型列表 |

5.1.5.3 设备运行状态 (ServiceState_T)

| | |
|--|--|
| 定义 | |
| <pre>enum ServiceState_T { IN_SERVICE, OUT_OF_SERVICE, OUT_OF_SERVICE_BY_MAINTENANCE, SERV_NA };</pre> | |
| 说明 | |
| 对象说明 | 设备运行状态 |
| 类型取值 | 取值说明 |
| IN_SERVICE | 单元盘已经插入到槽位中, 并已经根据网管系统的配置方式投入运行 |
| OUT_OF_SERVICE | 表示该单元盘已经不能运行, 即不能根据网管系统的配置完成指定的功能, 而且不能执行相应的管理动作 |
| OUT_OF_SERVICE_BY_MAINTENANCE | 该单元盘由于维护的目的而处于非正常工作状态 |
| SERV_NA | 单元盘的状态不明确 |

5.1.5.4 设备容器类型 (EquipmentHolderType_T)

| 定义 | |
|---------------------------------------|--------|
| typedef string EquipmentHolderType_T; | |
| 说明 | |
| 对象说明 | 设备容器类型 |
| 类型取值 | 取值说明 |
| "rack" | 机架 |
| "shelf" | 子架 |
| "sub_shelf" | 子子架 |
| "slot" | 槽位 |
| "sub_slot" | 子槽位 |

5.1.5.5 设备容器状态 (HolderState_T)

| 定义 | |
|--|-----------------------|
| <pre>enum HolderState_T { EMPTY, INSTALLED_AND_EXPECTED, EXPECTED_AND_NOT_INSTALLED, INSTALLED_AND_NOT_EXPECTED, MISMATCH_OF_INSTALLED_AND_EXPECTED, UNAVAILABLE, UNKNOWN };</pre> | |
| 说明 | |
| 对象说明 | 设备容器状态 |
| 类型取值 | 取值说明 |
| EMPTY | 空状态 (网管未配置设备) |
| INSTALLED_AND_EXPECTED | 安装了设备且类型符合 |
| EXPECTED_AND_NOT_INSTALLED | 期待安装设备但未安装 (网管已配置了设备) |
| INSTALLED_AND_NOT_EXPECTED | 安装了设备但未期待 (网管未安装) |
| MISMATCH_OF_INSTALLED_AND_EXPECTED | 安装了设备但类型不符合 |
| UNAVAILABLE | 不可用 |
| UNKNOWN | 无关状态 |

5.1.5.6 设备 (Equipment_T)

| 定义 | |
|--|--|
| <pre>struct Equipment_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; boolean alarmReportingIndicator;</pre> | |

| | |
|------------------------------|---|
| ServiceState_T | serviceState; |
| EquipmentObjectType_T | expectedEquipmentObjectType; |
| EquipmentObjectType_T | installedEquipmentObjectType; |
| string | installedPartNumber; |
| string | installedVersion; |
| string | installedSerialNumber; |
| globaldefs::NVSList_T | additionalInfo; |
| }; | |
| 说明 | |
| 对象说明 | 设备（单元盘）信息 |
| 类型取值 | 取值说明 |
| name | 设备的名称 |
| userLabel | 设备的用户标签 |
| nativeEMSName | 设备的 EMS 本地名称 |
| owner | 设备的所有者 |
| alarmReportingIndicator | 设备的告警上报是否被激活 |
| serviceState | 设备的运行状态 |
| expectedEquipmentObjectType | 期待的设备类型，如果没有可填为空字符串 |
| installedEquipmentObjectType | 已安装的设备类型，如果没有可填为空字符串 |
| installedPartNumber | 安装设备的零件编号，如果没有可填为空字符串 |
| installedVersion | 安装设备的版本，包括软件版本和硬件版本，如果没有可填为空字符串 |
| installedSerialNumber | 安装设备的序列号，如果没有可填为空字符串 |
| additionalInfo | 附加信息。包括： <ul style="list-style-type: none"> — 子类型，包括 OTU 盘、交叉盘、OPM、WSS 等； — 是否有保护及保护方式； — 单元盘能够提供的端口数量 |

5.1.5.7 设备容器（EquipmentHolder_T）

| | |
|--|--|
| 定义 | |
| <pre> struct EquipmentHolder_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; boolean alarmReportingIndicator; EquipmentHolderType_T holderType; globaldefs::NamingAttributes_T expectedOrInstalledEquipment; EquipmentObjectTypeList_T acceptableEquipmentTypeList; HolderState_T holderState; globaldefs::NVSList_T additionalInfo; }; </pre> | |

| 说明 | |
|------------------------------|--|
| 对象说明 | 设备容器信息。设备容器指机架、机框和槽位 |
| 类型取值 | 取值说明 |
| name | 设备容器的名称 |
| userLabel | 设备容器的用户标签 |
| nativeEMSName | 设备容器的 EMS 本地名称 |
| owner | 设备容器的所有者 |
| alarmReportingIndicator | 设备容器的告警上报是否被激活 |
| holderType | 设备容器的类型。 |
| expectedOrInstalledEquipment | 应安板名称，或者已安板名称。如果是空字符串，表示无应安板 |
| acceptableEquipmentTypeList | 可以接受的子设备或者设备容器列表。如果是槽位和子槽位则是必须的，表示槽位可以安装的设备 |
| holderState | 设备容器当前的状态 |
| additionalInfo | 附加信息。包括： <ul style="list-style-type: none"> — 子类型；（可选） — 设备容器版本； — 设备容器所在位置（槽位在子架中的相对位置，序号表示）； — 设备容器供应商名称； — 设备容器序列号 |

5.1.5.8 设备与容器区分器（EquipmentTypeQualifier_T）

| 定义 | |
|---|------------------------------------|
| <pre>enum EquipmentTypeQualifier_T { EQT, EQT HOLDER };</pre> | |
| 说明 | |
| 对象说明 | 设备与容器区分器，在设备与设备容器的联合类型中用于区分设备和设备容器 |
| 类型取值 | 取值说明 |
| EQT | 设备 |
| EQT HOLDER | 设备容器 |

5.1.5.9 设备或设备容器（EquipmentOrHolder_T）

| 定义 | |
|---|--------------|
| <pre>union EquipmentOrHolder_T switch (EquipmentTypeQualifier_T) { case EQT: Equipment_T equip; case EQT HOLDER: EquipmentHolder_T holder; };</pre> | |
| 说明 | |
| 对象说明 | 设备容器与设备的联合结构 |
| 类型取值 | 取值说明 |
| equip | 设备 |
| holder | 设备容器 |

5.1.5.10 设备或设备容器列表 (EquipmentOrHolderList_T)

| | |
|--|-----------|
| 定义 | |
| typedef sequence<EquipmentOrHolder_T> EquipmentOrHolderList_T; | |
| 说明 | |
| 对象说明 | 设备或设备容器列表 |

5.1.5.11 EquipmentOrHolderIterator_I 接口

5.1.5.11.1 从迭代器中查询数据 (next_n)

| | |
|--|---|
| 定义 | |
| <pre>boolean next_n (in unsigned long how_many, out EquipmentOrHolderList_T equipmentOrHolderList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时，系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | equipmentOrHolderList: 首次查询返回的设备或设备容器列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.5.11.2 查询迭代器记录条数 (getLength)

| | |
|--|---|
| 定义 | |
| <pre>unsigned long getLength () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数(注意这里指的是迭代器中数据记录的总条数,该值在迭代器生命周期内是不变的) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.5.11.3 销毁迭代器对象 (destroy)

| | |
|---|----------------------|
| 定义 | |
| <pre>void destroy () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.5.12 EquipmentInventoryMgr_I 接口 (从 common::Common_I 继承)

5.1.5.12.1 开启告警上报开关 (setAlarmReportingOn)

| | |
|---|--|
| 定义 | |
| <pre>void setAlarmReportingOn (in globaldefs::NamingAttributes_T equipmentOrHolderName) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 开启告警上报开关 |
| 输入参数 | equipmentOrHolderName: 设备或设备容器名称 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS |

5.1.5.12.2 关闭告警上报开关 (setAlarmReportingOff)

| | |
|--|--|
| 定义 | |
| <pre>void setAlarmReportingOff (in globaldefs::NamingAttributes_T equipmentOrHolderName) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 关闭告警上报开关 |
| 输入参数 | equipmentOrHolderName: 设备或设备容器名称 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS |

5.1.5.12.3 查询被直接包含的设备或设备容器 (getContainedEquipment)

| | |
|--|-----------------|
| 定义 | |
| <pre>void getContainedEquipment (in globaldefs::NamingAttributes_T equipmentHolderName, out EquipmentOrHolderList_T equipmentOrHolderList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询被直接包含的设备或设备容器 |

| | |
|---------|--|
| 输入参数 | equipmentOrHolderName: 设备或设备容器名称 |
| 输入/输出参数 | 无 |
| 输出参数 | equipmentOrHolderList: 设备或设备容器列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS |

5.1.5.12.4 查询指定设备信息 (getEquipment)

| | |
|--|---|
| 定义 | |
| <pre>void getEquipment (in globaldefs::NamingAttributes_T equipmentOrHolderName, out EquipmentOrHolder_T equip) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询指定设备信息 |
| 输入参数 | equipmentOrHolderName: 设备或设备容器名称 |
| 输入/输出参数 | 无 |
| 输出参数 | equip: 设备或容器信息 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.5.12.5 查询网元下所有设备或设备容器信息 (getAllEquipment)

| | |
|--|--|
| 定义 | |
| <pre>void getAllEquipment (in globaldefs::NamingAttributes_T meOrHolderName, in unsigned long how_many, out EquipmentOrHolderList_T eqList, out EquipmentOrHolderIterator_I eqIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询网元下所有设备或设备容器信息 |
| 输入参数 | meOrHolderName: 网元或设备容器名称。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | eqList: 设备或设备容器列表。 eqIt: 迭代查询设备或设备容器接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.5.12.6 查询网元下所有设备或设备容器名称 (getAllEquipmentNames)

| 定义 | |
|--|--|
| <pre>void getAllEquipmentNames (in globaldefs::NamingAttributes_T meOrHolderName, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询网元下所有设备或设备容器名称 |
| 输入参数 | meOrHolderName: 网元或设备容器名称。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 迭代查询名称接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.5.12.7 查询 TP 所在的设备 (getAllSupportingEquipment)

| 定义 | |
|--|--|
| <pre>void getAllSupportingEquipment (in globaldefs::NamingAttributes_T ptpOrMfdName, out EquipmentOrHolderList_T eqList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询 TP 所在的设备 |
| 输入参数 | ptpOrMfdName: TP 或 MFD 名称 |
| 输入/输出参数 | 无 |
| 输出参数 | eqList: 设备列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.5.12.8 查询 TP 所在的设备名称 (getAllSupportingEquipmentNames)

| 定义 | |
|--|--|
| <pre>void getAllSupportingEquipmentNames (in globaldefs::NamingAttributes_T ptpOrMfdName, out globaldefs::NamingAttributesList_T nameList) raises (globaldefs::ProcessingFailureException);</pre> | |

| 说明 | |
|---------|--|
| 功能描述 | 查询 TP 所在的设备名称 |
| 输入参数 | ptpOrMfdName: TP 或 MFD 名称 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.5.12.9 查询设备上支持的 PTP (getAllSupportedPTPs)

| 定义 | |
|---|--|
| <pre>void getAllSupportedPTPs (in globaldefs::NamingAttributes_T equipmentName, in unsigned long how_many, out terminationPoint::TerminationPointList_T tpList, out terminationPoint::TerminationPointIterator_I tpIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询设备上支持的 PTP |
| 输入参数 | equipmentName: 设备名称。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | tpList: 查询获得的终端点列表。 tpIt: 迭代查询终端点接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.5.12.10 查询设备上支持的 PTP 名称 (getAllSupportedPTPNames)

| 定义 | |
|---|---|
| <pre>void getAllSupportedPTPNames (in globaldefs::NamingAttributes_T equipmentName, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询设备上支持的 PTP 名称 |
| 输入参数 | equipmentName: 设备名称。 how_many: 首次迭代查询返回的数据数目 |

| | |
|---------|--|
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 查询获得的终端点名称列表。 nameIt: 迭代查询名称接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.6 终端点模块 (module terminationPoint)

5.1.6.1 方向类型 (Directionality_T)

| | |
|---|---------|
| 定义 | |
| <pre>enum Directionality_T { D_NA, D_BIDIRECTIONAL, D_SOURCE, D_SINK };</pre> | |
| 说明 | |
| 对象说明 | 终端点方向类型 |
| 类型取值 | 取值说明 |
| D_NA | 无关 |
| D_BIDIRECTIONAL | 双向 |
| D_SOURCE | 源 |
| D_SINK | 宿 |

5.1.6.2 TP 连接状态 (TPConnectionState_T)

| | |
|---|---------------------------|
| 定义 | |
| <pre>enum TPConnectionState_T { TPCS_NA, TPCS_SOURCE_CONNECTED, TPCS_SINK_CONNECTED, TPCS_BI_CONNECTED, TPCS_NOT_CONNECTED };</pre> | |
| 说明 | |
| 对象说明 | 终端点连接状态 |
| 类型取值 | 取值说明 |
| TPCS_NA | 无关 |
| TPCS_SOURCE_CONNECTED | 终端点的 SOURCE 部分参与了连接 |
| TPCS_SINK_CONNECTED | 终端点的 SINK 部分参与了连接 |
| TPCS_BI_CONNECTED | 终端点的 SINK 和 SOURCE 都参与了连接 |
| TPCS_NOT_CONNECTED | 未连接 |

5.1.6.3 TP 类型 (TPType_T)

| 定义 | |
|--|-------|
| <pre>enum TPType_T { TPT_PTP, TPT_CTP, TPT_TPPool };</pre> | |
| 说明 | |
| 对象说明 | 终端点类型 |
| 类型取值 | 取值说明 |
| TPT_PTP | 物理终端点 |
| TPT_CTP | 连接终端点 |
| TPT_TPPool | 终端点池 |

5.1.6.4 TP 映射模式 (TerminationMode_T)

| 定义 | |
|--|------------------------------|
| <pre>enum TerminationMode_T { TM_NA, TM_NEITHER_TERMINATED_NOR_AVAILABLE_FOR_MAPPING, TM_TERMINATED_AND_AVAILABLE_FOR_MAPPING };</pre> | |
| 说明 | |
| 对象说明 | TP 映射模式 |
| 类型取值 | 取值说明 |
| TM_NA | 无关 |
| TM_NEITHER_TERMINATED_NOR_AVAILABLE_FOR_MAPPING | 终端点可以配置交叉连接，但终端点不能进一步展开使用 |
| TM_TERMINATED_AND_AVAILABLE_FOR_MAPPING | 终端点的下层终端点可以使用，但终端点本身不能配置交叉连接 |

5.1.6.5 TP 保护相关类型 (TPProtectionAssociation_T)

| 定义 | |
|--|-------------------------|
| <pre>enum TPProtectionAssociation_T { TPPA_NA, TPPA_PSR_RELATED };</pre> | |
| 说明 | |
| 对象说明 | TP 保护相关类型 |
| 类型取值 | 取值说明 |
| TPPA_NA | 无关 |
| TPPA_PSR_RELATED | 与保护相关，即存在与终端点具有保护关系的终端点 |

5.1.6.6 TP (TerminationPoint_T)

| 定义 | |
|---|---|
| <pre> struct TerminationPoint_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; globaldefs::NamingAttributes_T ingressTrafficDescriptorName; globaldefs::NamingAttributes_T egressTrafficDescriptorName; TPType_T type; TPConnectionState_T connectionState; TerminationMode_T tpMappingMode; Directionality_T direction; transmissionParameters::LayeredParameterList_T transmissionParams; TPProtectionAssociation_T tpProtectionAssociation; boolean edgePoint; globaldefs::NVSList_T additionalInfo; }; </pre> | |
| 说明 | |
| 对象说明 | TP 信息 |
| 属性名 | 属性说明 |
| name | TP 名称 |
| userLabel | TP 的用户标签 |
| nativeEMSName | TP 的本地名称 |
| owner | TP 的所有者 |
| ingressTrafficDescriptorName | 入业务描述符名称 |
| egressTrafficDescriptorName | 出业务描述符名称 |
| type | TP 的类型 |
| connectionState | TP 的连接状态 |
| tpMappingMode | TP 的映射类型 |
| direction | TP 的方向 |
| transmissionParams | 层速率及传送参数列表。OTN 相关的层参数，见本标准第 4 部分 4.1 的表 1 OTN 层参数定义 |
| tpProtectionAssociation | 保护相关类型 |
| edgePoint | 是否是子网内部拓扑连接的终结点 |
| additionalInfo | 附加信息。包括： — 物理终端点序号（在所属单元盘下唯一） |

5.1.6.7 TP 列表 (TerminationPointList_T)

| 定义 | |
|---|-------|
| <pre> typedef sequence<TerminationPoint_T> TerminationPointList_T; </pre> | |
| 说明 | |
| 对象说明 | TP 列表 |

5.1.6.8 TerminationPointIterator_I 接口

5.1.6.8.1 从迭代器中查询数据 (next_n)

| 定义 | |
|--|---|
| <pre>boolean next_n (in unsigned long how_many, out TerminationPointList_T tpList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时，系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many : 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | $tpList$: 首次查询返回的终端点列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.6.8.2 查询迭代器记录条数 (getLength)

| 定义 | |
|--|---|
| <pre>unsigned long getLength () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数 (注意这里指的是迭代器中数据记录的总条数, 该值在迭代器生命周期内是不变的) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.6.8.3 销毁迭代器对象 (destroy)

| 定义 | |
|---|----------------------|
| <pre>void destroy () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.7 保护模块 (module protection)

5.1.7.1 保护策略的锁定状态 (ProtectionSchemeState_T)

| 定义 | |
|---|-----------|
| <pre>enum ProtectionSchemeState_T { PSS_UNKNOWN, PSS_AUTOMATIC, PSS_FORCED_OR_LOCKED_OUT };</pre> | |
| 说明 | |
| 对象说明 | 保护策略的锁定状态 |
| 类型取值 | 取值说明 |
| PSS_UNKNOWN | 无关 |
| PSS_AUTOMATIC | 部分锁定 |
| PSS_FORCED_OR_LOCKED_OUT | 整个保护组被锁定 |

5.1.7.2 保护类型 (ProtectionType_T)

| 定义 | |
|---|--------|
| <pre>enum ProtectionType_T { PT_MSP_APS, PT_SNCP };</pre> | |
| 说明 | |
| 对象说明 | 保护类型 |
| 类型取值 | 取值说明 |
| PT_MSP_APS | 复用段保护 |
| PT_SNCP | SNC 保护 |

5.1.7.3 倒换原因 (SwitchReason_T)

| 定义 | |
|---|------|
| <pre>enum SwitchReason_T { SR_NA, SR_RESTORED, SR_SIGNAL_FAIL, SR_SIGNAL_MISMATCH, SR_SIGNAL_DEGRADE, SR_AUTOMATIC_SWITCH, SR_MANUAL };</pre> | |
| 说明 | |
| 对象说明 | 倒换原因 |
| 类型取值 | 取值说明 |
| SR_NA | 无关 |
| SR_RESTORED | 恢复 |
| SR_SIGNAL_FAIL | 信号失效 |
| SR_SIGNAL_MISMATCH | 信号适配 |
| SR_SIGNAL_DEGRADE | 信号裂化 |
| SR_AUTOMATIC_SWITCH | 自动倒换 |
| SR_MANUAL | 人工倒换 |

5.1.7.4 设备倒换原因 (ESwitchReason_T) (可选)

| | |
|---------------------------------|---|
| 定义 | |
| typedef string ESwitchReason_T; | |
| 说明 | |
| 对象说明 | 设备倒换原因。 "SR_NA"表示未知。 "SR_E_FAILURE"表示设备失效。 "SR_MANUAL"表示人工倒换 |

5.1.7.5 保护倒换命令类型 (ProtectionCommand_T)

| | |
|--|----------|
| 定义 | |
| enum ProtectionCommand_T { PC_CLEAR, PC_LOCKOUT, PC_FORCED_SWITCH, PC_MANUAL_SWITCH, PC_EXERCISER }; | |
| 说明 | |
| 对象说明 | 保护倒换命令类型 |
| 类型取值 | 取值说明 |
| PC_CLEAR | 清除 |
| PC_LOCKOUT | 锁定倒换 |
| PC_FORCED_SWITCH | 强制倒换 |
| PC_MANUAL_SWITCH | 人工倒换 |
| PC_EXERCISER | 练习倒换 |

5.1.7.6 保护组类型 (ProtectionGroupType_T)

| | |
|--|---------|
| 定义 | |
| enum ProtectionGroupType_T { PGT_MSP_1_PLUS_1, PGT_MSP_1_FOR_N, PGT_2_FIBER_BLSR, PGT_4_FIBER_BLSR }; | |
| 说明 | |
| 对象说明 | 保护组类型 |
| 类型取值 | 取值说明 |
| PGT_MSP_1_PLUS_1 | 1+1 保护 |
| PGT_MSP_1_FOR_N | 1: N 保护 |
| PGT_2_FIBER_BLSR | 二纤环保护 |
| PGT_4_FIBER_BLSR | 四纤环保护 |

5.1.7.7 设备保护组类型 (EProtectionGroupType_T) (可选)

| | |
|--|---------|
| 定义 | |
| typedef string EProtectionGroupType_T; | |
| 说明 | |
| 对象说明 | 设备保护组类型 |

5.1.7.8 恢复模式 (ReversionMode_T)

| | |
|--|------|
| 定义 | |
| enum ReversionMode_T { RM_UNKNOWN, RM_NON_REVERTIVE, RM_REVERTIVE }; | |
| 说明 | |
| 对象说明 | 恢复模式 |
| 类型取值 | 取值说明 |
| RM_UNKNOWN | 不清楚 |
| RM_NON_REVERTIVE | 不恢复 |
| RM_REVERTIVE | 自动恢复 |

5.1.7.9 保护组 (ProtectionGroup_T)

| | |
|--|----------|
| 定义 | |
| struct ProtectionGroup_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; ProtectionGroupType_T protectionGroupType; ProtectionSchemeState_T protectionSchemeState; ReversionMode_T reversionMode; transmissionParameters::LayerRate_T rate; globaldefs::NamingAttributesList_T pgpTPList; globaldefs::NVSList_T pgpParameters; globaldefs::NVSList_T additionalInfo; }; | |
| 说明 | |
| 对象说明 | 保护组信息 |
| 属性名 | 属性说明 |
| name | 保护组名称 |
| userLabel | 保护组的用户标签 |
| nativeEMSName | 保护组的本地名称 |
| owner | 保护组的所有者 |
| protectionGroupType | 保护组类型 |

| | |
|-----------------------|--|
| protectionSchemeState | 保护方案的锁定状态 |
| reversionMode | 恢复模式 |
| rate | 速率 |
| pgpTPList | 构成保护组的 TP 列表, 保护 TP 标识和被保护 TP 标识的有序列表 |
| pgpParameters | 保护组参数, 见本标准第 4 部分 4.9.5 的表 2 “OTN 保护组参数定义” |
| additionalInfo | 附加信息。包括: — 子类型 (可选) |

5.1.7.10 保护组列表 (ProtectionGroupList_T)

| | |
|---|-------|
| 定义 | |
| typedef sequence <ProtectionGroup_T> ProtectionGroupList_T; | |
| 说明 | |
| 对象说明 | 保护组列表 |

5.1.7.11 设备保护组 (EProtectionGroup_T) (可选)

| | |
|--|-------------|
| 定义 | |
| <pre> struct EProtectionGroup_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; EProtectionGroupType_T eProtectionGroupType; ProtectionSchemeState_T protectionSchemeState; ReversionMode_T reversionMode; globaldefs::NamingAttributesList_T protectedList; globaldefs::NamingAttributesList_T protectingList; globaldefs::NVSList_T ePgpParameters; globaldefs::NVSList_T additionalInfo; }; </pre> | |
| 说明 | |
| 对象说明 | 设备保护组信息 |
| 属性名 | 属性说明 |
| name | 设备保护组名称 |
| userLabel | 设备保护组的用户标签 |
| nativeEMSName | 设备保护组的本地名称 |
| owner | 设备保护组的所有者 |
| eProtectionGroupType | 设备保护组类型 |
| protectionSchemeState | 保护方案的锁定状态 |
| reversionMode | 恢复模式 |
| protectedList | 被保护的设备列表 |
| protectingList | 保护其他设备的设备列表 |
| ePgpParameters | 设备保护组参数 |
| additionalInfo | 附加信息 |

5.1.7.12 设备保护组列表 (EProtectionGroupList_T) (可选)

| | |
|---|---------|
| 定义 | |
| typedef sequence <EProtectionGroup_T> EProtectionGroupList_T; | |
| 说明 | |
| 对象说明 | 设备保护组列表 |

5.1.7.13 倒换数据 (SwitchData_T)

| | |
|--|---------|
| 定义 | |
| <pre> struct SwitchData_T { ProtectionType_T protectionType; SwitchReason_T switchReason; transmissionParameters::LayerRate_T layerRate; globaldefs::NamingAttributes_T groupName; globaldefs::NamingAttributes_T protectedTP; globaldefs::NamingAttributes_T switchToTP; globaldefs::NVList_T additionalInfo; }; </pre> | |
| 说明 | |
| 对象说明 | 倒换数据 |
| 属性名 | 属性说明 |
| protectionType | 保护类型 |
| switchReason | 倒换原因 |
| layerRate | 层速率 |
| groupName | 保护组名称 |
| protectedTP | 被保护的 TP |
| switchToTP | 倒换到的 TP |
| additionalInfo | 附加信息 |

5.1.7.14 倒换数据列表 (SwitchDataList_T)

| | |
|--|--------|
| 定义 | |
| typedef sequence<SwitchData_T> SwitchDataList_T; | |
| 说明 | |
| 对象说明 | 倒换数据列表 |

5.1.7.15 设备倒换数据 (ESwitchData_T) (可选)

| | |
|---|--|
| 定义 | |
| <pre> struct ESwitchData_T { EProtectionGroupType_T eProtectionGroupType; ESwitchReason_T eSwitchReason; globaldefs::NamingAttributes_T ePGPName; globaldefs::NamingAttributes_T protectedE; globaldefs::NamingAttributes_T switchToE; globaldefs::NVList_T additionalInfo; }; </pre> | |

| 说明 | |
|----------------------|---------|
| 对象说明 | 设备倒换数据 |
| 属性名 | 属性说明 |
| eProtectionGroupType | 设备保护类型 |
| eSwitchReason | 倒换原因 |
| ePGPName | 设备保护组名称 |
| protectedE | 被保护的设备 |
| switchToE | 倒换到的设备 |
| additionalInfo | 附加信息 |

5.1.7.16 设备倒换数据列表 (ESwitchDataList_T) (可选)

| 定义 | |
|--|----------|
| typedef sequence<ESwitchData_T> ESwitchDataList_T; | |
| 说明 | |
| 对象说明 | 设备倒换数据列表 |

5.1.7.17 ProtectionGroupIterator_I 接口

5.1.7.17.1 从迭代器中查询数据 (next_n)

| 定义 | |
|--|---|
| boolean next_n (in unsigned long how_many, out ProtectionGroupList_T pgpList) raises (globaldefs::ProcessingFailureException); | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时，系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | pgpList: 首次查询返回的保护组列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.7.17.2 查询迭代器记录条数 (getLength)

| 定义 | |
|--|---|
| unsigned long getLength () raises (globaldefs::ProcessingFailureException); | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数(注意这里指的是迭代器中数据记录的总条数,该值在迭代器生命周期内是不变的) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.7.17.3 销毁迭代器对象 (destroy)

| 定义 | |
|---|----------------------|
| void destroy () raises (globaldefs::ProcessingFailureException); | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.7.18 EProtectionGroupIterator_I 接口 (可选)

5.1.7.18.1 从迭代器中查询数据 (next_n)

| 定义 | |
|---|--|
| boolean next_n (in unsigned long how_many, out EProtectionGroupList_T eGPList) raises (globaldefs::ProcessingFailureException); | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时, 系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | eGPList: 首次查询返回的设备保护组列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.7.18.2 查询迭代器记录条数 (getLength)

| 定义 | |
|--|---|
| unsigned long getLength () raises (globaldefs::ProcessingFailureException); | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数 (注意这里指的是迭代器中数据记录的总条数, 该值在迭代器生命周期内是不变的) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.7.18.3 销毁迭代器对象 (destroy)

| 定义 | |
|---|--|
| void destroy () raises (globaldefs::ProcessingFailureException); | |

| 说明 | |
|---------|----------------------|
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.7.19 ProtectionMgr_I 接口（从 common::Common_I 继承）

5.1.7.19.1 查询网元下所有保护组信息（getAllProtectionGroups）

| 定义 | |
|---|--|
| <pre>void getAllProtectionGroups (in globaldefs::NamingAttributes_T meName, in unsigned long how_many, out ProtectionGroupList_T pgList, out ProtectionGroupIterator_I pgpIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询网元下所有保护组信息。 |
| 输入参数 | meName: 网元名称。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | pgList: 保护组列表。 pgpIt: 保护组迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.7.19.2 查询网元下所有设备保护组信息（getAllEProtectionGroups）（可选）

| 定义 | |
|---|---|
| <pre>void getAllEProtectionGroups (in globaldefs::NamingAttributes_T meName, in unsigned long how_many, out EProtectionGroupList_T epgpList, out EProtectionGroupIterator_I epgpIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询网元下所有设备保护组信息 |
| 输入参数 | meName: 网元名称。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | epgpList: 设备保护组列表。 epgpIt: 设备保护组迭代查询接口 |

| | |
|------|--|
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |
|------|--|

5.1.7.19.3 查询指定的保护组信息 (getProtectionGroup)

| | |
|--|---|
| 定义 | |
| <pre>void getProtectionGroup (in globaldefs::NamingAttributes_T pgName, out protection::ProtectionGroup_T protectionGroup) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询指定的保护组信息 |
| 输入参数 | pgName: 保护组名称 |
| 输入/输出参数 | 无 |
| 输出参数 | protectionGroup: 保护组 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.7.19.4 查询指定的设备保护组信息 (getEProtectionGroup) (可选)

| | |
|--|---|
| 定义 | |
| <pre>void getEProtectionGroup (in globaldefs::NamingAttributes_T ePGName, out protection::EProtectionGroup_T eProtectionGroup) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询指定的设备保护组信息。 |
| 输入参数 | pgName: 保护组名称 |
| 输入/输出参数 | 无 |
| 输出参数 | protectionGroup: 保护组 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.7.19.5 查询所有的保护功能已被封锁的不可抢占的通道 TP 名称 (getAllNUTTPNames) (可选)

| | |
|--|--|
| 定义 | |
| <pre>void getAllNUTTPNames (in globaldefs::NamingAttributes_T pgName, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException);</pre> | |

| 说明 | |
|---------|---|
| 功能描述 | 查询所有的保护功能已被封锁的不可抢占的通道 TP 名称 |
| 输入参数 | pgName: 保护组名称 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 迭代查询名称接口 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.7.19.6 查询保护组中可配置为抢占式的额外路径的终端点 (getAllPreemptibleTPNames) (可选)

| 定义 | |
|--|---|
| <pre>void getAllPreemptibleTPNames (in globaldefs::NamingAttributes_T pgName, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询保护组中可配置为抢占式的额外路径的终端点 |
| 输入参数 | pgName: 保护组名称。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 迭代查询名称接口 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.7.19.7 查询所有被保护的 TP 名称 (getAllProtectedTPNames) (可选)

| 定义 | |
|--|--|
| <pre>void getAllProtectedTPNames (in globaldefs::NamingAttributes_T pgName, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException);</pre> | |

| 说明 | |
|---------|---|
| 功能描述 | 查询所有被保护的 TP 名称 |
| 输入参数 | pgName: 保护组名称。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 迭代查询名称接口 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.7.19.8 查询保护倒换数据 (retrieveSwitchData)

| 定义 | |
|--|--|
| <pre>void retrieveSwitchData (in globaldefs::NamingAttributes_T reliableSinkCtpOrGroupName, out protection::SwitchDataList_T switchData) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 指定对象名称, 查询该对象的保护倒换数据 |
| 输入参数 | reliableSinkCtpOrGroupName: 保护组名称或宿 CTP 名称 |
| 输入/输出参数 | 无 |
| 输出参数 | switchData: 保护倒换数据列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.7.19.9 查询设备保护倒换数据 (retrieveESwitchData) (可选)

| 定义 | |
|---|--|
| <pre>void retrieveESwitchData (in globaldefs::NamingAttributes_T ePGPName, out protection::ESwitchDataList_T eSwitchDataList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 指定对象名称, 查询该对象的设备保护倒换数据 |
| 输入参数 | ePGPName: 设备保护组名称 |
| 输入/输出参数 | 无 |
| 输出参数 | eSwitchDataList: 设备保护倒换数据列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.7.19.10 执行保护倒换命令 (performProtectionCommand)

| 定义 | |
|---|--|
| <pre>void performProtectionCommand (in ProtectionCommand_T protectionCommand, in globaldefs::NamingAttributes_T reliableSinkCtpOrGroupName, in globaldefs::NamingAttributes_T fromTp, in globaldefs::NamingAttributes_T toTp, out protection::SwitchData_T switchData) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 执行保护倒换命令 |
| 输入参数 | protectionCommand: 保护倒换命令。 reliableSinkCtpOrGroupName: 保护组名称或宿 CTP 名称。 fromTp: 被倒换的对象名称。 toTp: 倒换后到达的对象名称 |
| 输入/输出参数 | 无 |
| 输出参数 | switchData: 倒换后的保护倒换状态 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS |

5.1.8 子网模块 (module multiLayerSubnetwork)

5.1.8.1 子网的拓扑类型 (Topology_T)

| 定义 | |
|---|----------|
| <pre>enum Topology_T { TOPO_SINGLETON, TOPO_CHAIN, TOPO_PSR, TOPO_OPEN_PSR, TOPO_SPRING, TOPO_OPEN_SPRING, TOPO_MESH };</pre> | |
| 说明 | |
| 对象说明 | 子网的拓扑类型 |
| 类型取值 | 取值说明 |
| TOPO_SINGLETON | 简单节点 |
| TOPO_CHAIN | 线型 |
| TOPO_PSR | 通道倒换环 |
| TOPO_OPEN_PSR | 开放的通道倒换环 |
| TOPO_SPRING | 共享环 |
| TOPO_OPEN_SPRING | 开放的共享环 |
| TOPO_MESH | 网孔型 |

5.1.8.2 EMS 的操作自由度 (EMSFreedomLevel_T)

| 定义 | |
|---|--|
| <pre>enum EMSFreedomLevel_T { EMSFL_CC_AT_SNC_LAYER, EMSFL_TERMINATE_AND_MAP, EMSFL_HIGHER_ORDER_SNCS, EMSFL_RECONFIGURATION };</pre> | |
| 说明 | |
| 对象说明 | EMS 的操作自由度 |
| 类型取值 | 取值说明 |
| EMSFL_CC_AT_SNC_LAYER | EMS 可以创建或删除可被和已被 SNC 使用的交叉连接 |
| EMSFL_TERMINATE_AND_MAP | 在 EMSFL_CC_AT_SNC_LAYER 基础上, EMS 可以映射去映射或通道化去通道化可被和已被 SNC 使用的终端点 |
| EMSFL_HIGHER_ORDER_SNCS | 在 EMSFL_TERMINATE_AND_MAP 基础上, EMS 可以创建或删除可被或已被用作承载 SNC 的高阶交叉连接 |
| EMSFL_RECONFIGURATION | EMS 可以进行任何操作, 即 NMS 不限制 EMS 进行与 SNC 相关的操作 |

5.1.8.3 多层子网 (MultiLayerSubnetwork_T)

| 定义 | |
|---|-------------|
| <pre>struct MultiLayerSubnetwork_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; Topology_T subnetworkType; transmissionParameters::LayerRateList_T supportedRates; globaldefs::NVSList_T additionalInfo; };</pre> | |
| 说明 | |
| 对象说明 | 多层子网信息 |
| 属性名 | 属性说明 |
| name | 多层子网名称 |
| userLabel | 多层子网的用户标签 |
| nativeEMSName | 多层子网的本地名称 |
| owner | 多层子网的所有者 |
| subnetworkType | 多层子网类型 |
| supportedRates | 多层子网支持的业务速率 |
| additionalInfo | 附加信息 |

5.1.8.4 多层子网列表 (SubnetworkList_T)

| 定义 | |
|---|--------|
| <pre>typedef sequence<MultiLayerSubnetwork_T> SubnetworkList_T;</pre> | |
| 说明 | |
| 对象说明 | 多层子网列表 |

5.1.8.5 SubnetworkIterator_1 接口

5.1.8.5.1 从迭代器中查询数据 (next_n)

| 定义 | |
|--|---|
| <pre>boolean next_n (in unsigned long how_many, out SubnetworkList_T subnetworkList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时，系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | subnetworkList: 首次查询返回的多层子网列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.8.5.2 查询迭代器记录条数 (getLength)

| 定义 | |
|--|---|
| <pre>unsigned long getLength () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数(注意这里指的是迭代器中数据记录的总条数,该值在迭代器生命周期内是不变的) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.8.5.3 销毁迭代器对象 (destroy)

| 定义 | |
|---|----------------------|
| <pre>void destroy () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.8.6 MultiLayerSubnetworkMgr_I 接口 (从 common::Common_I 继承)

5.1.8.6.1 查询子网下所有的网元信息 (getAllManagedElements)

| 定义 | |
|--|--|
| <pre>void getAllManagedElements (in globaldefs::NamingAttributes_T subnetName, in unsigned long how_many, out managedElement::ManagedElementList_T meList, out managedElement::ManagedElementIterator_I meIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询子网下所有的网元信息 |
| 输入参数 | subnetName: 子网名称。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | meList: 网元列表。 meIt: 网元迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.8.6.2 查询子网下所有的网元名称 (getAllManagedElementNames)

| 定义 | |
|--|--|
| <pre>void getAllManagedElementNames (in globaldefs::NamingAttributes_T subnetName, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询子网下所有的网元名称 |
| 输入参数 | subnetName: 子网名称。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 名称迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.8.6.3 查询多层子网 (getMultiLayerSubnetwork)

| 定义 | |
|--|---|
| <pre>void getMultiLayerSubnetwork (in globaldefs::NamingAttributes_T subnetName, out MultiLayerSubnetwork_T subnetwork) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询多层子网 |
| 输入参数 | subnetName: 子网名称 |
| 输入/输出参数 | 无 |
| 输出参数 | subnetwork: 多层子网信息 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.8.6.4 查询子网下所有的拓扑连接 (getAllTopologicalLinks)

| 定义 | |
|---|--|
| <pre>void getAllTopologicalLinks (in globaldefs::NamingAttributes_T subnetName, in unsigned long how_many, out topologicalLink::TopologicalLinkList_T topoList, out topologicalLink::TopologicalLinkIterator_I topoIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询子网下所有的拓扑连接 |
| 输入参数 | subnetName: 子网名称。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | topoList: 拓扑连接列表。 topoIt: 拓扑连接迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.8.6.5 查询子网下所有的拓扑连接名称 (getAllTopologicalLinkNames)

| 定义 | |
|---|--|
| <pre>void getAllTopologicalLinkNames (in globaldefs::NamingAttributes_T subnetName, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |

| 说明 | |
|---------|--|
| 功能描述 | 查询子网下所有的拓扑连接名称 |
| 输入参数 | subnetName: 子网名称。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 名称迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.8.6.6 查询指定的拓扑连接 (getTopologicalLink)

| 定义 | |
|--|---|
| <pre>void getTopologicalLink (in globaldefs::NamingAttributes_T topoLinkName, out topologicalLink::TopologicalLink_T topoLink) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询拓扑连接 |
| 输入参数 | topoLinkName: 拓扑连接名称 |
| 输入/输出参数 | 无 |
| 输出参数 | topoLink: 拓扑连接信息 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.8.6.7 查询子网下所有的边界点 (getAllEdgePoints) (可选)

| 定义 | |
|--|---|
| <pre>void getAllEdgePoints (in globaldefs::NamingAttributes_T subnetName, in transmissionParameters::LayerRateList_T tpLayerRateList, in transmissionParameters::LayerRateList_T connectionLayerRateList, in unsigned long how_many, out terminationPoint::TerminationPointList_T tpList, out terminationPoint::TerminationPointIterator_I tpIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询子网下所有的边界点 |
| 输入参数 | subnetName: 子网名称。 tpLayerRateList: TP点包含的速率参数列表 connectionLayerRateList: 连接速率列表。 how_many: 首次迭代查询返回的数据数目 |

| | |
|---------|--|
| 输入/输出参数 | 无 |
| 输出参数 | tpList: TP 列表。 tpIt: TP 迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.8.6.8 查询子网下所有的边界点名称 (getAllEdgePointNames) (可选)

| | |
|--|--|
| 定义 | |
| <pre>void getAllEdgePointNames (in globaldefs::NamingAttributes_T subnetName, in transmissionParameters::LayerRateList_T layerRateList, in transmissionParameters::LayerRateList_T connectionLayerRateList, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询子网下所有的边界点名称 |
| 输入参数 | subnetName: 子网名称。 tpLayerRateList: TP 点包含的速率参数列表。 connectionLayerRateList: 连接速率列表。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 名称迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.8.6.9 查询保护相关 TP (getAssociatedTP) (可选)

| | |
|---|-----------|
| 定义 | |
| <pre>void getAssociatedTP (in globaldefs::NamingAttributes_T tpName, out terminationPoint::TerminationPointList_T tpList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询保护相关 TP |

| | |
|---------|--|
| 输入参数 | tpName: TP 名称 |
| 输入/输出参数 | 无 |
| 输出参数 | tpList: TP 列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.8.6.10 查询所有的 SNC (getAllSubnetworkConnections)

| | |
|--|--|
| 定义 | |
| <pre>void getAllSubnetworkConnections (in globaldefs::NamingAttributes_T subnetName, in transmissionParameters::LayerRateList_T connectionRateList, in unsigned long how_many, out subnetworkConnection::SubnetworkConnectionList_T sncList, out subnetworkConnection::SNCIterator_I sncIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询所有的 SNC |
| 输入参数 | subnetName: 子网名称。 connectionLayerRateList: 连接速率列表。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | sncList: SNC 列表。 sncIt: SNC 迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.8.6.11 查询所有的 SNC 名称 (getAllSubnetworkConnectionNames)

| | |
|---|--------------|
| 定义 | |
| <pre>void getAllSubnetworkConnectionNames (in globaldefs::NamingAttributes_T subnetName, in transmissionParameters::LayerRateList_T connectionRateList, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询所有的 SNC 名称 |

| | |
|---------|--|
| 输入参数 | subnetName: 子网名称。 connectionLayerRateList: 连接速率列表。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 名称迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.8.6.12 查询指定 TP 下的所有 SNC (getAllSubnetworkConnectionsWithTP)

| | |
|---|--|
| 定义 | |
| <pre>void getAllSubnetworkConnectionsWithTP (in globaldefs::NamingAttributes_T tpName, in transmissionParameters::LayerRateList_T connectionRateList, in unsigned long how_many, out subnetworkConnection::SubnetworkConnectionList_T sncList, out subnetworkConnection::SNCIterator_I sncIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询指定 TP 下的所有 SNC |
| 输入参数 | tpName: TP 名称。 connectionLayerRateList: 连接速率列表。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | sncList: SNC 列表。 sncIt: SNC 迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.8.6.13 查询 SNC 路由 (getRoute)

| | |
|---|--|
| 定义 | |
| <pre>void getRoute (in globaldefs::NamingAttributes_T sncName, in boolean includeHigherOrderCCs, out subnetworkConnection::Route_T route) raises (globaldefs::ProcessingFailureException);</pre> | |

| 说明 | |
|---------|--|
| 功能描述 | 指定 SNC 名称，查询 SNC 路由信息 |
| 输入参数 | sncName: SNC 名称; includeHigherOrderCCs: 是否返回承载 SNC 的路由的高阶交叉连接 |
| 输入/输出参数 | 无 |
| 输出参数 | route: SNC 的路由 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_NOT_IMPLEMENTED EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY |

5.1.8.6.14 查询指定 TP 下的所有 SNC 名称 (getAllSubnetworkConnectionNamesWithTP)

| 定义 | |
|--|--|
| <pre>void getAllSubnetworkConnectionNamesWithTP (in globaldefs::NamingAttributes_T tpName, in transmissionParameters::LayerRateList_T connectionRateList, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询指定 TP 下的所有 SNC 名称 |
| 输入参数 | tpName: TP 名称。 connectionLayerRateList: 连接速率列表。 how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 名称列表。 nameIt: 名称迭代查询接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.1.8.6.15 查询指定的 SNC (getSNC)

| 定义 | |
|--|-----------|
| <pre>void getSNC (in globaldefs::NamingAttributes_T sncName, out subnetworkConnection::SubnetworkConnection_T snc) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询指定的 SNC |

| | |
|---------|---|
| 输入参数 | sncName: SNC 名称 |
| 输入/输出参数 | 无 |
| 输出参数 | snc: SNC 信息 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.8.6.16 通过用户标签查询 SNC (getSNCsByUserLabel) (可选)

| | |
|--|--|
| 定义 | |
| <pre>void getSNCsByUserLabel (in string userLabel, out subnetworkConnection::SubnetworkConnectionList_T sncList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 通过用户标签查询 SNC |
| 输入参数 | userLabel: 子网用户标签 |
| 输入/输出参数 | 无 |
| 输出参数 | sncList: SNC 列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_UNABLE_TO_COMPLY EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS |

5.1.8.6.17 创建 SNC (createSNC)

| | |
|---|---|
| 定义 | |
| <pre>void createSNC (in subnetworkConnection::SNCCreateData_T createData, in subnetworkConnection::GradesOfImpact_T tolerableImpact, in EMSFreedomLevel_T emsFreedomLevel, out subnetworkConnection::SubnetworkConnection_T theSNC, out string errorReason) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 创建 SNC |
| 输入参数 | createData: 子网连接创建数据。 tolerableImpact: 可容忍的业务影响级别。 emsFreedomLevel: EMS 操作自由度级别 |
| 输入/输出参数 | 无 |
| 输出参数 | theSNC: 创建成功的 SNC。 errorReason: 错误原因 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT |

| | |
|--|---|
| | EXCPT_ENTITY_NOT_FOUND EXCPT_PROTECTION_EFFORT_NOT_MET EXCPT_UNABLE_TO_COMPLY EXCPT_UNSUPPORTED_ROUTING_CONSTRAINTS EXCPT_USERLABEL_IN_USE EXCPT_OBJECT_IN_USE |
|--|---|

5.1.8.6.18 激活 SNC (activateSNC)

| 定义 | |
|---|---|
| <pre>void activateSNC (in globaldefs::NamingAttributes_T sncName, in subnetworkConnection::GradesOfImpact_T tolerableImpact, in EMSFreedomLevel_T emsFreedomLevel, inout subnetworkConnection::TPDataList_T tpsToModify, out subnetworkConnection::SubnetworkConnection_T theSNC, out string errorReason) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 激活 SNC |
| 输入参数 | sncName: SNC 名称。 tolerableImpact: 可容忍的业务影响级别。 emsFreedomLevel: EMS 操作自由度级别 |
| 输入/输出参数 | tpsToModify: 修改的 TP 列表 |
| 输出参数 | theSNC: 激活后的 SNC。 errorReason: 错误原因 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_PROTECTION_EFFORT_NOT_MET EXCPT_UNABLE_TO_COMPLY EXCPT_UNSUPPORTED_ROUTING_CONSTRAINTS EXCPT_USERLABEL_IN_USE EXCPT_OBJECT_IN_USE EXCPT_NE_COMM_LOSS EXCPT_NOT_IN_VALID_STATE |

5.1.8.6.19 创建并激活 SNC (createAndActivateSNC)

| 定义 | |
|---|--|
| <pre>void createAndActivateSNC (in subnetworkConnection::SNCCreateData_T createData, in subnetworkConnection::GradesOfImpact_T tolerableImpact, in EMSFreedomLevel_T emsFreedomLevel, inout subnetworkConnection::TPDataList_T tpsToModify,</pre> | |

| | |
|--|---|
| <pre> out subnetworkConnection::SubnetworkConnection_T theSNC, out string errorReason) raises (globaldefs::ProcessingFailureException); </pre> | |
| 说明 | |
| 功能描述 | 创建并激活 SNC |
| 输入参数 | createData: 子网连接创建数据。 tolerableImpact: 可容忍的业务影响级别。 emsFreedomLevel: EMS 操作自由度级别 |
| 输入/输出参数 | tpsToModify: 修改的 TP 列表 |
| 输出参数 | theSNC: 创建并激活后的 SNC。 errorReason: 错误原因 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_PROTECTION_EFFORT_NOT_MET EXCPT_UNABLE_TO_COMPLY EXCPT_UNSUPPORTED_ROUTING_CONSTRAINTS EXCPT_USERLABEL_IN_USE EXCPT_OBJECT_IN_USE EXCPT_NE_COMM_LOSS EXCPT_NOT_IN_VALID_STATE |

5.1.8.6.20 去激活 SNC (deactivateSNC)

| | |
|---|--|
| 定义 | |
| <pre> void deactivateSNC (in globaldefs::NamingAttributes_T sncName, in subnetworkConnection::GradesOfImpact_T tolerableImpact, in EMSFreedomLevel_T emsFreedomLevel, inout subnetworkConnection::TPDataList_T tpsToModify, out subnetworkConnection::SubnetworkConnection_T theSNC, out string errorReason) raises (globaldefs::ProcessingFailureException); </pre> | |
| 说明 | |
| 功能描述 | 去激活 SNC |
| 输入参数 | sncName: SNC 名称。 tolerableImpact: 可容忍的业务影响级别。 emsFreedomLevel: EMS 操作自由度级别 |
| 输入/输出参数 | tpsToModify: 修改的 TP 列表 |
| 输出参数 | theSNC: 去激活后的 SNC。 errorReason: 错误原因 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR |

| | |
|--|--|
| | EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_PROTECTION_EFFORT_NOT_MET EXCPT_UNABLE_TO_COMPLY EXCPT_UNSUPPORTED_ROUTING_CONSTRAINTS EXCPT_USERLABEL_IN_USE EXCPT_OBJECT_IN_USE EXCPT_NE_COMM_LOSS EXCPT_NOT_IN_VALID_STATE |
|--|--|

5.1.8.6.21 删除 SNC (deleteSNC)

| | |
|---|--|
| 定义 | |
| <pre>void deleteSNC (in globaldefs::NamingAttributes_T sncName, in EMSFreedomLevel_T emsFreedomLevel) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 删除 SNC |
| 输入参数 | sncName: SNC 名称。 emsFreedomLevel: EMS 操作自由度级别 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NOT_IN_VALID_STATE EXCPT_UNABLE_TO_COMPLY |

5.1.8.6.22 去激活并删除 SNC (deactivateAndDeleteSNC)

| | |
|---|--|
| 定义 | |
| <pre>void deactivateAndDeleteSNC (in globaldefs::NamingAttributes_T sncName, in subnetworkConnection::GradesOfImpact_T tolerableImpact, in EMSFreedomLevel_T emsFreedomLevel, inout subnetworkConnection::TPDataList_T tpsToModify, out subnetworkConnection::SubnetworkConnection_T theSNC, out string errorReason) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 去激活并删除 SNC |
| 输入参数 | sncName: SNC 名称。 tolerableImpact: 可容忍的业务影响级别。 emsFreedomLevel: EMS 操作自由度级别 |

| | |
|---------|---|
| 输入/输出参数 | tpsToModify: 修改的 TP 列表 |
| 输出参数 | theSNC: 去激活并删除的 SNC。 errorReason: 错误原因 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS |

5.1.8.6.23 检查 SNC 创建数据是否有效 (checkValidSNC) (可选)

| 定义 | |
|---|---|
| <pre>void checkValidSNC (in subnetworkConnection::SNCCreateData_T createData, in subnetworkConnection::TPDataList_T tpsToModify, in boolean considerResources, out boolean valid) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 检查 SNC 创建数据是否有效 |
| 输入参数 | createData: SNC 创建数据。 tpsToModify: TP 修改数据。 considerResources: 是否考虑资源分配 |
| 输入/输出参数 | tpsToModify: 修改的 TP 列表 |
| 输出参数 | valid: 是否有效 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_OBJECT_IN_USE EXCPT_INVALID_INPUT EXCPT_NE_COMM_LOSS EXCPT_UNSUPPORTED_ROUTING_CONSTRAINTS EXCPT_USERLABEL_IN_USE EXCPT_UNABLE_TO_COMPLY EXCPT_OBJECT_IN_USE |

5.1.8.6.24 修改 SNC (modifySNC)

| 定义 | |
|---|--|
| <pre>void modifySNC (in globaldefs::NamingAttributes_T sncName, in string routeId, in subnetworkConnection::SNCModifyData_T SNCModifyData, in subnetworkConnection::GradesOfImpact_T tolerableImpact, in subnetworkConnection::ProtectionEffort_T tolerableImpactEffort, in EMSFreedomLevel_T emsFreedomLevel, inout subnetworkConnection::TPDataList_T tpsToModify,</pre> | |

| | |
|--|--|
| <pre> out subnetworkConnection::SubnetworkConnection_T newSNC, out string errorReason) raises (globaldefs::ProcessingFailureException); </pre> | |
| 说明 | |
| 功能描述 | 修改 SNC |
| 输入参数 | sncName: SNC 名称。 routeId: 路由标识。 SNCModifyData: SNC 修改数据。 tolerableImpact: 可容忍的业务影响级别。 tolerableImpactEffort: 可容忍的保护尽力程度。 emsFreedomLevel: EMS 操作自由度级别 |
| 输入/输出参数 | tpsToModify: 修改的 TP 列表 |
| 输出参数 | newSNC: 修改后的 SNC。 errorReason: 错误原因 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_OBJECT_IN_USE EXCPT_ENTITY_NOT_FOUND EXCPT_UNSUPPORTED_ROUTING_CONSTRAINTS EXCPT_USERLABEL_IN_USE EXCPT_UNABLE_TO_COMPLY EXCPT_OBJECT_IN_USE |

5.1.9 子网连接模块 (module subnetworkConnection)

5.1.9.1 静态保护级别 (StaticProtectionLevel_T)

| | |
|---|--------|
| 定义 | |
| <pre> enum StaticProtectionLevel_T { PREEMPTIBLE, UNPROTECTED, PARTIALLY_PROTECTED, FULLY_PROTECTED, HIGHLY_PROTECTED }; </pre> | |
| 说明 | |
| 对象说明 | 静态保护级别 |
| 类型取值 | 取值说明 |
| PREEMPTIBLE | 可抢占 |
| UNPROTECTED | 无保护 |
| PARTIALLY_PROTECTED | 部分保护 |
| FULLY_PROTECTED | 完全保护 |
| HIGHLY_PROTECTED | 高级保护 |

5.1.9.2 保护尽力程度 (ProtectionEffort_T)

| 定义 | |
|---|---------------|
| <pre>enum ProtectionEffort_T { EFFORT_WHATEVER, EFFORT_SAME_OR_BETTER, EFFORT_SAME_OR_WORSE, EFFORT_SAME };</pre> | |
| 说明 | |
| 对象说明 | 保护尽力程度 |
| 类型取值 | 取值说明 |
| EFFORT_WHATEVER | 其他保护级别也可以 |
| EFFORT_SAME_OR_BETTER | 可选择相同或更好的保护级别 |
| EFFORT_SAME_OR_WORSE | 可选择相同或差的保护级别 |
| EFFORT_SAME | 选择相同的保护级别 |

5.1.9.3 子网连接状态 (SNCSState_T)

| 定义 | |
|--|--------|
| <pre>enum SNCSState_T { SNCS_NONEXISTENT, SNCS_PENDING, SNCS_ACTIVE, SNCS_PARTIAL };</pre> | |
| 说明 | |
| 对象说明 | 子网连接状态 |
| 类型取值 | 取值说明 |
| SNCS_NONEXISTENT | 不存在 |
| SNCS_PENDING | 挂起 |
| SNCS_ACTIVE | 激活 |
| SNCS_PARTIAL | 部分激活 |

5.1.9.4 业务影响级别 (GradesOfImpact_T)

| 定义 | |
|---|--------|
| <pre>enum GradesOfImpact_T { GOI_HITLESS, GOI_MINOR_IMPACT, GOI_MAJOR_IMPACT };</pre> | |
| 说明 | |
| 对象说明 | 业务影响级别 |
| 类型取值 | 取值说明 |
| GOI_HITLESS | 无影响 |
| GOI_MINOR_IMPACT | 次要影响 |
| GOI_MAJOR_IMPACT | 主要影响 |

5.1.9.5 TP 数据 (TPData_T)

| 定义 | |
|---|-----------|
| <pre>struct TPData_T { globaldefs::NamingAttributes_T tpName; terminationPoint::TerminationMode_T tpMappingMode; transmissionParameters::LayeredParameterList_T transmissionParams; globaldefs::NamingAttributes_T ingressTrafficDescriptorName; globaldefs::NamingAttributes_T egressTrafficDescriptorName; };</pre> | |
| 说明 | |
| 对象说明 | 终端点数据 |
| 类型取值 | 取值说明 |
| tpName | TP 名称 |
| tpMappingMode | TP 映射方式 |
| transmissionParams | 层参数列表 |
| ingressTrafficDescriptorName | 入口业务描述符名称 |
| egressTrafficDescriptorName | 出口业务描述符名称 |

5.1.9.6 TP 数据列表 (TPDataList_T)

| 定义 | |
|---|---------|
| <pre>typedef sequence<TPData_T> TPDataList_T;</pre> | |
| 说明 | |
| 对象说明 | TP 数据列表 |

5.1.9.7 SNC 类型 (SNCType_T)

| 定义 | |
|--|---------------|
| <pre>enum SNCType_T { ST_SIMPLE, ST_ADD_DROP_A, ST_ADD_DROP_Z, ST_INTERCONNECT, ST_DOUBLE_INTERCONNECT, ST_DOUBLE_ADD_DROP, ST_OPEN_ADD_DROP, ST_EXPLICIT };</pre> | |
| 说明 | |
| 对象说明 | SNC 类型 |
| 类型取值 | 取值说明 |
| ST_SIMPLE | 简单型 |
| ST_ADD_DROP_A | A 点 AddDrop 型 |
| ST_ADD_DROP_Z | Z 点 AddDrop 型 |
| ST_INTERCONNECT | 互连型 |

| | |
|------------------------|---------------|
| ST_DOUBLE_INTERCONNECT | 双互连型 |
| ST_DOUBLE_ADD_DROP | 双向 AddDrop 型 |
| ST_OPEN_ADD_DROP | 开放的 AddDrop 型 |
| ST_EXPLICIT | 其他型 |

5.1.9.8 重路由允许类型 (Reroute_T)

| | |
|---|---------|
| 定义 | |
| <pre>enum Reroute_T { RR_NA, RR_NO, RR_YES };</pre> | |
| 说明 | |
| 对象说明 | 重路由允许类型 |
| 类型取值 | 取值说明 |
| RR_NA | 无关 |
| RR_NO | 不允许 |
| RR_YES | 允许 |

5.1.9.9 网络层重路由允许类型 (NetworkRouted_T)

| | |
|---|------------|
| 定义 | |
| <pre>enum NetworkRouted_T { NR_NA, NR_NO, NR_YES };</pre> | |
| 说明 | |
| 对象说明 | 网络层重路由允许类型 |
| 类型取值 | 取值说明 |
| NR_NA | 无关 |
| NR_NO | 不允许 |
| NR_YES | 允许 |

5.1.9.10 SNC (SubnetworkConnection_T)

| | |
|---|--|
| 定义 | |
| <pre>struct SubnetworkConnection_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; SNCState_T sncState; globaldefs::ConnectionDirection_T direction; transmissionParameters::LayerRate_T rate; StaticProtectionLevel_T staticProtectionLevel; };</pre> | |

| | |
|---|---------------|
| <pre> SNCType_T sncType; TPDataList_T aEnd; TPDataList_T zEnd; Reroute_T rerouteAllowed; NetworkRouted_T networkRouted; globaldefs::NVSList_T additionalInfo; }; </pre> | |
| 说明 | |
| 对象说明 | SNC 信息 |
| 属性名 | 属性说明 |
| name | SNC 名称 |
| userLabel | SNC 的用户标签 |
| nativeEMSName | SNC 的本地名称 |
| owner | SNC 的所有者 |
| sncState | SNC 的状态 |
| direction | SNC 的方向 |
| rate | SNC 的速率 |
| staticProtectionLevel | SNC 的静态保护等级 |
| sncType | SNC 类型 |
| aEnd | SNC 的 A 点名称列表 |
| zEnd | SNC 的 Z 点名称列表 |
| rerouteAllowed | 重路由允许 |
| networkRouted | 网络层重路由允许 |
| additionalInfo | 附加信息 |

5.1.9.11 SNC 列表 (SubnetworkConnectionList_T)

| | |
|---|--------|
| 定义 | |
| <pre> typedef sequence<SubnetworkConnection_T> SubnetworkConnectionList_T; </pre> | |
| 说明 | |
| 对象说明 | SNC 列表 |

5.1.9.12 交叉连接 (CrossConnect_T)

| | |
|--|--------|
| 定义 | |
| <pre> struct CrossConnect_T { boolean active; globaldefs::ConnectionDirection_T direction; SNCType_T ccType; globaldefs::NamingAttributesList_T aEndNameList; globaldefs::NamingAttributesList_T zEndNameList; globaldefs::NVSList_T additionalInfo; }; </pre> | |
| 说明 | |
| 对象说明 | 交叉连接信息 |
| 属性名 | 属性说明 |

| | |
|----------------|---------------------------------------|
| active | 交叉连接的激活状态 |
| direction | 交叉连接的方向 |
| ccType | 交叉连接的类型。 |
| aEndNameList | 交叉连接的 A 点名称列表 |
| zEndNameList | 交叉连接的 Z 点名称列表 |
| additionalInfo | 附加信息。包括： — 网元标识符； — 交叉连接标识符（可选） |

5.1.9.13 交叉连接列表 (CrossConnectList_T)

| | |
|--|--------|
| 定义 | |
| typedef sequence<CrossConnect_T> CrossConnectList_T; | |
| 说明 | |
| 对象说明 | 交叉连接列表 |

5.1.9.14 资源信息 (Resource_T)

| | |
|--|------|
| 定义 | |
| typedef globaldefs::NamingAttributes_T Resource_T; | |
| 说明 | |
| 对象说明 | 资源信息 |

5.1.9.15 资源信息列表 (ResourceList_T)

| | |
|--|-----------------------------------|
| 定义 | |
| typedef sequence<Resource_T> ResourceList_T; | |
| 说明 | |
| 对象说明 | 资源信息列表，包含网元、拓扑连接、物理终端点、连接终端点、子网连接 |

5.1.9.16 路由信息 (Route_T)

| | |
|---|------|
| 定义 | |
| typedef sequence<CrossConnect_T> Route_T; | |
| 说明 | |
| 对象说明 | 路由信息 |

5.1.9.17 路由描述符信息 (RouteDescriptor_T)

| | |
|--|--|
| 定义 | |
| <pre> struct RouteDescriptor_T { string id; string intended; string actualState; string administrativeState; string inUseBy; string exclusive; Route_T routeXCs; globaldefs::NVSList_T additionalInfo; }; </pre> | |

| 说明 | |
|---------------------|--|
| 对象说明 | 路由描述符信息 |
| 属性名 | 属性说明 |
| id | 标识符 |
| intended | 是否为期望路由, 取值: "y", "n" |
| actualState | 状态, 取值: "inactive"表示包含的交叉连接未激活; "active": 表示包含的交叉连接已激活; "partial": 表示包含的交叉连接部分激活 |
| administrativeState | 管理状态, 取值"locked"表示路由不能被激活; "unlocked"表示路由可以被激活 |
| inUseBy | 是否已被其他连接使用, 取值: "y", "n" |
| exclusive | 是否独占, 取值: "y", "n" |
| routeXCs | 包含的交叉连接 |
| additionalInfo | 附加信息 |

5.1.9.18 子网连接路由列表 (RouteList_T)

| 定义 | |
|--|----------|
| typedef sequence<RouteDescriptor_T> RouteList_T; | |
| 说明 | |
| 对象说明 | 子网连接路由列表 |

5.1.9.19 SNC 创建数据 (SNCCreateData_T)

| 定义 | |
|--|-----------|
| <pre> struct SNCCreateData_T { string userLabel; boolean forceUniqueness; string owner; globaldefs::ConnectionDirection_T direction; StaticProtectionLevel_T staticProtectionLevel; ProtectionEffort_T protectionEffort; Reroute_T rerouteAllowed; NetworkRouted_T networkRouted; SNCType_T sncType; transmissionParameters::LayerRate_T layerRate; CrossConnectList_T ccInclusions; ResourceList_T neTpInclusions; boolean fullRoute; ResourceList_T neTpSncExclusions; globaldefs::NamingAttributesList_T aEnd; globaldefs::NamingAttributesList_T zEnd; globaldefs::NVSList_T additionalCreationInfo; }; </pre> | |
| 说明 | |
| 对象说明 | SNC 创建数据 |
| 属性名 | 属性说明 |
| userLabel | SNC 的用户标签 |

| | |
|------------------------|---------------|
| forceUniqueness | 是否要求用户标签唯一 |
| owner | SNC 的所有者 |
| direction | SNC 的方向 |
| staticProtectionLevel | SNC 的静态保护等级 |
| protectionEffort | 保护尽力程度 |
| rerouteAllowed | 重路由允许 |
| networkRouted | 网络层重路由允许 |
| sncType | SNC 类型 |
| layerRate | 层速率 |
| ccInclusions | 必须包含的交叉连接 |
| neTpInclusions | 必须包含的资源 |
| fullRoute | 是否提供的完整路由 |
| neTpSncExclusions | 必须排除的资源 |
| aEnd | SNC 的 A 点名称列表 |
| zEnd | SNC 的 Z 点名称列表 |
| additionalCreationInfo | 附加信息 |

5.1.9.20 SNC 创建数据列表 (SNCCreateDataList_T)

| | |
|--|------------|
| 定义 | |
| typedef sequence<SNCCreateData_T> SNCCreateDataList_T; | |
| 说明 | |
| 对象说明 | SNC 创建数据列表 |

5.1.9.21 SNC 修改数据 (SNCMModifyData_T)

| | |
|---|--|
| 定义 | |
| <pre> struct SNCMModifyData_T { string userLabel; boolean forceUniqueness; string owner; globaldefs::ConnectionDirection_T direction; string modifyType; boolean retainOldSNC; boolean modifyServers_allowed; StaticProtectionLevel_T staticProtectionLevel; ProtectionEffort_T protectionEffort; Reroute_T rerouteAllowed; NetworkRouted_T networkRouted; SNCType_T sncType; transmissionParameters::LayerRate_T layerRate; RouteList_T addedOrNewRoute; RouteList_T removedRoute; ResourceList_T neTpInclusions; boolean fullRoute; ResourceList_T neTpSncExclusions; globaldefs::NamingAttributesList_T aEnd; </pre> | |

| | |
|--|--|
| <pre>globaldefs::NamingAttributesList_T zEnd; globaldefs::NVSList_T additionalCreationInfo; };</pre> | |
| 说明 | |
| 对象说明 | SNC 修改数据 |
| 属性名 | 属性说明 |
| userLabel | SNC 的用户标签 |
| forceUniqueness | 是否要求用户标签唯一 |
| owner | SNC 的所有者 |
| direction | SNC 的方向 |
| modifyType | 修改类型, 包括: “rerouting”“add_protection”“remove_protection” |
| retainOldSNC | 是否请求 EMS 保留原来的 SNC 在挂起状态 |
| modifyServers_allowed | 是否允许修改服务层来实现保护限制 |
| staticProtectionLevel | SNC 的静态保护等级 |
| protectionEffort | 保护尽力程度 |
| rerouteAllowed | 重路由允许 |
| networkRouted | 网络层重路由允许 |
| sncType | SNC 类型 |
| layerRate | 层速率 |
| addedOrNewRoute | 增加或新建的路由 |
| removedRoute | 删除的路由 |
| neTpInclusions | 必须包含的资源 |
| fullRoute | 是否提供的完整路由 |
| neTpSncExclusions | 必须排除的资源 |
| aEnd | SNC 的 A 点名称列表 |
| zEnd | SNC 的 Z 点名称列表 |
| additionalCreationInfo | 附加信息 |

5.1.9.22 CClterator_I 接口

5.1.9.22.1 从迭代器中查询数据 (next_n)

| | |
|--|--|
| 定义 | |
| <pre>boolean next_n (in unsigned long how_many, out CrossConnectList_T ccList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时, 系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | ccList: 首次查询返回的交叉连接列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.9.22.2 查询迭代器记录条数 (getLength)

| 定义 | |
|--|---|
| <pre>unsigned long getLength () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数(注意这里指的是迭代器中数据记录的总条数,该值在迭代器生命周期内是不变的) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.9.22.3 销毁迭代器对象 (destroy)

| 定义 | |
|---|----------------------|
| <pre>void destroy () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.9.23 SNCIterator_I 接口

5.1.9.23.1 从迭代器中查询数据 (next_n)

| 定义 | |
|---|---|
| <pre>boolean next_n (in unsigned long how_many, out SubnetworkConnectionList_T sncList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时,系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | sncList: 首次查询返回的 SNC 列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.9.23.2 查询迭代器记录条数 (getLength)

| 定义 | |
|--|--|
| <pre>unsigned long getLength () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |

| | |
|---------|---|
| 功能描述 | 查询迭代器中数据记录的总条数（注意这里指的是迭代器中数据记录的总条数，该值在迭代器生命周期内是不变的） |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.1.9.23.3 销毁迭代器对象（destroy）

| | |
|--|----------------------|
| 定义 | |
| <pre>void destroy () raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.2 故障管理模块

OTN新增的故障数据，应符合YD/T2149.2第8章的要求。其他的故障数据应满足本节要求。

5.2.1 告警上报

告警上报数据类型见公共管理部分的通知管理模块。

5.2.2 告警级别模块（module aSAP）

5.2.2.1 分配级别（AssignedSeverity_T）

| | |
|--|--------|
| 定义 | |
| <pre>enum AssignedSeverity_T { AS_INDETERMINATE, AS_CRITICAL, AS_MAJOR, AS_MINOR, AS_WARNING, AS_NONALARMED, AS_FREE_CHOICE };</pre> | |
| 说明 | |
| 对象说明 | 分配级别 |
| 类型取值 | 取值说明 |
| AS_INDETERMINATE | 未确认 |
| AS_CRITICAL | 严重告警 |
| AS_MAJOR | 主要告警 |
| AS_MINOR | 次要告警 |
| AS_WARNING | 警告 |
| AS_FREE_CHOICE | 原始告警级别 |

5.2.2.2 告警级别分配 (AlarmSeverityAssignment_T)

| 定义 | |
|--|---|
| <pre>struct AlarmSeverityAssignment_T { notifications::ProbableCauseList_T probableCause; string probableCauseQualifier; string nativeProbableCause; AssignedSeverity_T serviceAffecting; AssignedSeverity_T serviceNonAffecting; AssignedSeverity_T serviceIndependentOrUnknown; };</pre> | |
| 说明 | |
| 对象说明 | 告警级别分配 |
| 属性名 | 属性说明 |
| probableCause | 告警原因列表 |
| probableCauseQualifier | 告警原因的唯一名称, 当告警原因不能保证唯一性时, 用来唯一标识告警级别信息 |
| nativeProbableCause | 告警在 EMS 本地的原因, 当告警原因不能保证唯一性时, 也可以用来唯一标识告警级别信息 |
| serviceAffecting | 影响服务 |
| serviceNonAffecting | 不影响服务 |
| serviceIndependentOrUnknown | 未知 |

5.2.2.3 告警级别分配列表 (AlarmSeverityAssignmentList_T)

| 定义 | |
|---|------------|
| <pre>typedef sequence<AlarmSeverityAssignment_T> AlarmSeverityAssignmentList_T;</pre> | |
| 说明 | |
| 对象说明 | 告警级别分配信息列表 |

5.2.2.4 告警级别分配模板 (ASAP_T) (可选)

| 定义 | |
|--|---------------|
| <pre>struct ASAP_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; boolean notModifiable; AlarmSeverityAssignmentList_T alarmSeverityAssignmentList; globaldefs::NVSLIST_T additionalInfo; };</pre> | |
| 说明 | |
| 对象说明 | 告警级别分配模板 |
| 属性名 | 属性说明 |
| name | 告警级别分配模板名称 |
| userLabel | 告警级别分配模板的用户标签 |

| | |
|-----------------------------|-------------------------------|
| nativeEMSName | 告警级别分配模板的本地 EMS 名称 |
| owner | 告警级别分配模板的所有者名称 |
| notModifiable | 是否不可修改。取值为“真”，表示告警级别是固定的，不能修改 |
| alarmSeverityAssignmentList | 告警级别分配列表 |
| additionalInfo | 附加信息 |

5.2.2.5 告警级别分配模板列表 (ASAPList_T) (可选)

| | |
|--------------------------------------|------------|
| 定义 | |
| typedef sequence<ASAP_T> ASAPList_T; | |
| 说明 | |
| 对象说明 | 告警级别分配模板列表 |

5.2.2.6 告警级别分配创建和修改数据 (ASAPCreateModifyData_T)

| | |
|---|---------------|
| 定义 | |
| <pre>struct ASAPCreateModifyData_T { string userLabel; boolean forceUniqueness; string owner; AlarmSeverityAssignmentList_T alarmSeverityAssignmentList; globaldefs::NVSList_T additionalInfo; };</pre> | |
| 说明 | |
| 对象说明 | 告警级别分配创建和修改数据 |
| 属性名 | 属性说明 |
| userLabel | 告警级别分配表的用户标签。 |
| forceUniqueness | 是否要求用户标签唯一 |
| owner | 告警级别分配表的所有者 |
| alarmSeverityAssignmentList | 告警级别分配列表 |
| additionalInfo | 附加信息 |

5.2.2.7 ASAPIterator_I 接口 (可选)

5.2.2.7.1 从迭代器中查询数据 (next_n)

| | |
|--|---|
| 定义 | |
| <pre>boolean next_n (in unsigned long how_many, out ASAPList_T aSAPList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时，系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | aSAPList: 首次查询返回的告警级别分配模板列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.2.2.7.2 查询迭代器记录条数 (getLength)

| | |
|--|---|
| 定义 | |
| unsigned long getLength () raises (globaldefs::ProcessingFailureException) ; | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数 (注意这里指的是迭代器中数据记录的总条数, 该值在迭代器生命周期内是不变的) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.2.2.7.3 销毁迭代器对象 (destroy)

| | |
|---|----------------------|
| 定义 | |
| void destroy () raises (globaldefs::ProcessingFailureException) ; | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.3 性能管理模块

5.3.1 性能管理模块 (module performance)

5.3.1.1 性能采集粒度 (Granularity_T)

| | |
|-------------------------------|---|
| 定义 | |
| typedef string Granularity_T; | |
| 说明 | |
| 对象说明 | 表示性能数据采集的粒度周期 有效值: "15min": 表示粒度周期为 15 分钟; "24h": 表示粒度周期为 24 小时; "NA": 表示当前的即时值 |

5.3.1.2 性能采集粒度周期列表 (GranularityList_T)

| | |
|---|-------------|
| 定义 | |
| typedef sequence <Granularity_T> GranularityList_T; | |
| 说明 | |
| 对象说明 | 性能采集粒度周期列表。 |

5.3.1.3 性能参数名称 (PMParameterName_T)

| | |
|-----------------------------------|--------|
| 定义 | |
| typedef string PMParameterName_T; | |
| 说明 | |
| 对象说明 | 性能参数名称 |

5.3.1.4 性能参数名称列表 (PMParameterNameList_T)

| | |
|--|----------|
| 定义 | |
| typedef sequence<PMParameterName_T> PMParameterNameList_T; | |
| 说明 | |
| 对象说明 | 性能参数名称列表 |

5.3.1.5 性能测量参数 (PMMeasurement_T)

| | |
|---|---|
| 定义 | |
| <pre>struct PMMeasurement_T { PMParameterName_T pmParameterName; PMLocation_T pmLocation; float value; string unit; string intervalStatus; };</pre> | |
| 说明 | |
| 对象说明 | 性能测量参数 |
| 属性名 | 属性说明 |
| pmParameterName | 性能参数名称 |
| pmLocation | 性能监测点位置 |
| value | 性能值 |
| unit | 性能值的单位 |
| intervalStatus | 表示描述性能值的有效性，允许的值有： "Valid": 表示数据有效； "Incomplete": 表示整个采集间隔（15分钟、24小时）内，数据不可用； "Invalid": 表示指定的时间间隔内，数据可用但标记为无效； "Unavailable": 表示指定的时间间隔内，数据完全不可用； "Zero-suppressed": 表示零抑制时间段 |

5.3.1.6 性能测量参数列表 (PMMeasurementList_T)

| | |
|---|----------|
| 定义 | |
| typedef sequence <PMMeasurement_T> PMMeasurementList_T; | |
| 说明 | |
| 对象说明 | 性能测量参数列表 |

5.3.1.7 性能数据 (PMDData_T)

| | |
|---|--|
| 定义 | |
| <pre>struct PMData_T { globaldefs::NamingAttributes_T tpName; transmissionParameters::LayerRate_T layerRate; Granularity_T granularity; globaldefs::Time_T retrievalTime; PMMeasurementList_T pmMeasurementList; };</pre> | |

| 说明 | |
|-------------------|----------|
| 对象说明 | 性能数据 |
| 属性名 | 属性说明 |
| tpName | 终端点名称 |
| layerRate | 层速率 |
| granularity | 性能采集粒度 |
| retrievalTime | 性能查询时间 |
| pmMeasurementList | 性能测量参数列表 |

5.3.1.8 性能数据列表 (PMDDataList_T)

| 定义 | |
|---|--------|
| typedef sequence<PMDData_T> PMDataList_T; | |
| 说明 | |
| 对象说明 | 性能数据列表 |

5.3.1.9 性能监测点位置 (PMLocation_T)

| 定义 | |
|------------------------------|---|
| typedef string PMLocation_T; | |
| 说明 | |
| 对象说明 | 表示性能监测的位置。 取值： "PML_NEAR_END_Rx": 表示近端接收； "PML_FAR_END_Rx": 表示远端接收； "PML_NEAR_END_Tx": 表示近端发送； "PML_FAR_END_Tx": 表示远端发送； "PML_BIDIRECTIONAL": 表示双向； "PML_CONTRA_NEAR_END_Rx": 表示反向的近端接收； "PML_CONTRA_FAR_END_Rx": 表示反向的远端接收 |

5.3.1.10 性能监测点位置列表 (PMLocationList_T)

| 定义 | |
|---|----------|
| typedef sequence <PMLocation_T> PMLocationList_T; | |
| 说明 | |
| 对象说明 | 性能监测位置列表 |

5.3.1.11 性能参数 (PMParameter_T)

| 定义 | |
|--|---------|
| <pre>struct PMParameter_T { PMParameterName_T pmParameterName; PMLocation_T pmLocation; };</pre> | |
| 说明 | |
| 对象说明 | 性能参数 |
| 属性名 | 属性说明 |
| pmParameterName | 性能参数名称 |
| pmLocation | 性能监测点位置 |

5.3.1.12 性能参数列表 (PMPParameterList_T)

| | |
|---|--------|
| 定义 | |
| typedef sequence <PMPParameter_T> PMPParameterList_T; | |
| 说明 | |
| 对象说明 | 性能参数列表 |

5.3.1.13 性能门限类型 (PMThresholdType_T)

| | |
|---|--------|
| 定义 | |
| <pre>enum PMThresholdType_T { TWM_HIGHEST, TWM_HIGH, TWM_LOW, TWM_LOWEST };</pre> | |
| 说明 | |
| 对象说明 | 性能门限类型 |
| 类型取值 | 取值说明 |
| TWM_HIGHEST | 最高限 |
| TWM_HIGH | 高限 |
| TWM_LOW | 低限 |
| TWM_LOWEST | 最低限 |

5.3.1.14 性能越限参数 (TCAPParameter_T)

| | |
|--|---------|
| 定义 | |
| <pre>struct TCAPParameter_T { PMPParameterName_T pmParameterName; Granularity_T granularity; PMLocation_T pmLocation; PMThresholdType_T thresholdType; boolean triggerFlag; float value; string unit; };</pre> | |
| 说明 | |
| 对象说明 | 性能越限参数 |
| 属性名 | 属性说明 |
| pmParameterName | 性能参数名称 |
| granularity | 性能采集粒度 |
| pmLocation | 性能监测点位置 |
| thresholdType | 性能门限类型 |
| triggerFlag | 激活标志 |
| value | 性能值 |
| unit | 性能值的单位 |

5.3.1.15 性能选择参数 (PMTPSelect_T)

| 定义 | |
|---|------------|
| <pre>struct PMTPSelect_T { globaldefs::NamingAttributes_T name; transmissionParameters::LayerRateList_T layerRateList; PMLocationList_T pMLocationList; GranularityList_T granularityList; };</pre> | |
| 说明 | |
| 对象说明 | 性能选择参数 |
| 属性名 | 属性说明 |
| name | 对象名称 |
| layerRateList | 速率层次列表 |
| pMLocationList | 性能监测点位置列表 |
| granularityList | 性能采集粒度周期列表 |

5.3.1.16 性能选择参数列表 (PMTPSelectList_T)

| 定义 | |
|---|----------|
| <pre>typedef sequence<PMTPSelect_T> PMTPSelectList_T;</pre> | |
| 说明 | |
| 对象说明 | 性能选择参数列表 |

5.3.1.17 性能门限值 (PMThreshold_T)

| 定义 | |
|--|---------|
| <pre>struct PMThreshold_T { PMParameterName_T pmParameterName; PMLocation_T pmLocation; PMThresholdType_T thresholdType; boolean triggerFlag; float value; string unit; };</pre> | |
| 说明 | |
| 对象说明 | 性能门限值 |
| 属性名 | 属性说明 |
| pmParameterName | 性能参数名称 |
| pmLocation | 性能监测点位置 |
| thresholdType | 性能门限类型 |
| triggerFlag | 激活标志 |
| value | 性能值 |
| unit | 性能值的单位 |

5.3.1.18 性能门限值列表 (PMThresholdList_T)

| | |
|---|---------|
| 定义 | |
| typedef sequence <PMThreshold_T> PMThresholdList_T; | |
| 说明 | |
| 对象说明 | 性能门限值列表 |

5.3.1.19 性能保持时间 (HoldingTime_T)

| | |
|---|-----------|
| 定义 | |
| <pre>struct HoldingTime_T { short storeTime24hr; short storeTime15min; };</pre> | |
| 说明 | |
| 对象说明 | 性能保持时间 |
| 属性名 | 属性说明 |
| storeTime24hr | 24 小时保持时间 |
| storeTime15min | 15 分钟保持时间 |

5.3.1.20 性能参数相关的门限 (PMParameterWithThresholds_T)

| | |
|--|-----------|
| 定义 | |
| <pre>struct PMParameterWithThresholds_T { PMParameterName_T pmParameterName; PMThresholdList_T pmThresholdList; };</pre> | |
| 说明 | |
| 对象说明 | 性能参数相关的门限 |
| 属性名 | 属性说明 |
| pmParameterName | 性能参数名称 |
| pmThresholdList | 性能门限列表 |

5.3.1.21 性能参数相关门限的列表 (PMParameterWithThresholdsList_T)

| | |
|---|-------------|
| 定义 | |
| typedef sequence <PMParameterWithThresholds_T> PMParameterWithThresholdsList_T; | |
| 说明 | |
| 对象说明 | 性能参数相关门限的列表 |

5.3.1.22 管理状态 (AdministrativeState_T)

| | |
|---|------|
| 定义 | |
| <pre>enum AdministrativeState_T { AS_Locked, AS_Unlocked };</pre> | |
| 说明 | |
| 对象说明 | 管理状态 |
| 类型取值 | 取值说明 |
| AS_Locked | 锁定 |
| AS_Unlocked | 激活 |

5.3.1.23 性能监测点 (PMP_T)

| 定义 | |
|---|-------------|
| <pre>struct PMP_T { globaldefs::NamingAttributes_T name; string userLabel; string nativeEMSName; string owner; PMPParameterWithThresholdsList_T pmParameterWithThresholdsList; AdministrativeState_T monitoringState; AdministrativeState_T supervisionState; globaldefs::NVSLList_T additionalInfo; };</pre> | |
| 说明 | |
| 对象说明 | 性能监测点 |
| 属性名 | 属性说明 |
| name | 性能监测点的名称 |
| userLabel | 性能监测点的用户标签 |
| nativeEMSName | 性能监测点的本地名称 |
| owner | 性能监测点的所有者 |
| pmParameterWithThresholdsList | 性能参数相关门限的列表 |
| monitoringState | 性能监测点的监测状态 |
| supervisionState | 性能监测点的管理状态 |
| additionalInfo | 附加信息 |

5.3.1.24 性能监测点列表 (PMPList_T)

| 定义 | |
|---|---------|
| <pre>typedef sequence<PMP_T> PMPList_T;</pre> | |
| 说明 | |
| 对象说明 | 性能监测点列表 |

5.3.1.25 性能数据文件目的地址 (Destination_T)

| 定义 | |
|--|------------|
| <pre>typedef string Destination_T;</pre> | |
| 说明 | |
| 对象说明 | 性能数据文件目的地址 |

5.3.1.26 PMDataIterator_I 接口

5.3.1.26.1 从迭代器中查询数据 (next_n)

| 定义 | |
|---|--|
| <pre>boolean next_n (in unsigned long how_many, out PMDataList_T pmDataList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时, 系统在数据取完后应该自动销毁该迭代器对象 |

| | |
|---------|---------------------------|
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | pmDataList: 首次查询返回的性能数据列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.3.1.26.2 查询迭代器记录条数 (getLength)

| | |
|--|---|
| 定义 | |
| <pre>unsigned long getLength () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数 (注意这里指的是迭代器中数据记录的总条数, 该值在迭代器生命周期内是不变的) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.3.1.26.3 销毁迭代器对象 (destroy)

| | |
|---|----------------------|
| 定义 | |
| <pre>void destroy () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.3.1.27 PMPIterator_I 接口

5.3.1.27.1 从迭代器中查询数据 (next_n)

| | |
|---|--|
| 定义 | |
| <pre>boolean next_n (in unsigned long how_many, out PMPList_T pmpList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时, 系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | pmpList: 首次查询返回的性能监测点列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.3.1.27.2 查询迭代器记录条数 (getLength)

| 定义 | |
|--|---|
| <pre>unsigned long getLength () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数(注意这里指的是迭代器中数据记录的总条数,该值在迭代器生命周期内是不变的) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 返回值 | unsigned long: 迭代器中包含的总的记录条数 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.3.1.27.3 销毁迭代器对象 (destroy)

| 定义 | |
|---|----------------------|
| <pre>void destroy () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.3.1.28 PerformanceManagementMgr_I 接口

5.3.1.28.1 查询网元性能的能力 (getMEPMCapabilities)

| 定义 | |
|--|--|
| <pre>void getMEPMcapabilities (in globaldefs::NamingAttributes_T meName, in transmissionParameters::LayerRate_T layerRate, out PMPParameterList_T pmParameterList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 指定对象名称, 查询网元性能的能力 |
| 输入参数 | meName: 指定的网元名称。 layerRate: 指定的速率层次 |
| 输入/输出参数 | 无 |
| 输出参数 | pmParameterList: 性能参数 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_NE_COMM_LOSS EXCPT_CAPACITY_EXCEEDED |

5.3.1.28.2 激活性能采集数据 (enablePMDData)

| | |
|--|--|
| 定义 | |
| <pre>void enablePMDData (in PMTPSelectList_T pmTPSelectList, out PMTPSelectList_T failedTPSelectList) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 激活指定性能监测点上的性能采集 |
| 输入参数 | pmTPSelectList: 性能选择参数列表 |
| 输入/输出参数 | 无 |
| 输出参数 | failedTPSelectList: 激活失败的性能选择参数列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_NE_COMM_LOSS EXCPT_CAPACITY_EXCEEDED |

5.3.1.28.3 去激活性能采集数据 (disablePMDData)

| | |
|---|---|
| 定义 | |
| <pre>void disablePMDData (in PMTPSelectList_T pmTPSelectList, out PMTPSelectList_T failedTPSelectList) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 去激活指定性能监测点上的性能采集 |
| 输入参数 | pmTPSelectList: 性能选择参数列表 |
| 输入/输出参数 | 无 |
| 输出参数 | failedTPSelectList: 去激活失败的性能选择参数列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_NE_COMM_LOSS |

5.3.1.28.4 清除性能寄存器 (clearPMDData)

| | |
|---|--|
| 定义 | |
| <pre>void clearPMDData (in PMTPSelectList_T pmTPSelectList, out PMTPSelectList_T failedTPSelectList) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 指定对象名称, 清除该对象的性能寄存器 |
| 输入参数 | pmTPSelectList: 性能选择参数列表 |
| 输入/输出参数 | 无 |
| 输出参数 | failedTPSelectList: 失败的性能选择参数列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_NE_COMM_LOSS |

5.3.1.28.5 查询所有的当前性能数据 (getAllCurrentPMDData)

| 定义 | |
|--|---|
| <pre>void getAllCurrentPMDData (in PMTPSelectList_T pmTPSelectList, in PMParameterNameList_T pmParameters, in unsigned long how_many, out PMDataList_T pmDataList, out PMDataIterator_I pmIt) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 指定对象名称和性能参数，查询所有的当前性能 |
| 输入参数 | <p>pmTPSelectList: 性能选择参数列表。</p> <p>pmParameters: 性能参数。</p> <p>how_many: 迭代查询数据方式下首次查询的数目</p> |
| 输入/输出参数 | 无 |
| 输出参数 | <p>pmDataList: 性能数据列表。</p> <p>pmIt: 迭代查询接口</p> |
| 操作异常 | <p>EXCPT_NOT_IMPLEMENTED</p> <p>EXCPT_INTERNAL_ERROR</p> <p>EXCPT_INVALID_INPUT</p> <p>EXCPT_NE_COMM_LOSS</p> <p>EXCPT_UNABLEO_COMPLY</p> |

5.3.1.28.6 查询历史性能数据 (getHistoryPMDData)

| 定义 | |
|--|--|
| <pre>void getHistoryPMDData (in Destination_T destination, in string userName, in string password, in PMTPSelectList_T pmTPSelectList, in PMParameterNameList_T pmParameters, in globaldefs::Time_T startTime, in globaldefs::Time_T endTime, in boolean forceUpload) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 指定对象名称和性能参数，将历史性能文件上载到指定的文件传输服务器 |
| 输入参数 | <p>destination: 文件传输服务器目标地址。</p> <p>userName: 文件传输用户名。</p> <p>password: 文件传输密码。</p> <p>pmTPSelectList: 性能选择参数列表。</p> <p>pmParameters: 性能参数。</p> <p>startTime: 历史性能起始时间。</p> <p>endTime: 历史性能结束时间。</p> <p>forceUpload: 是否强制全部上载</p> |

| | |
|---------|--|
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_NE_COMM_LOSS EXCPT_UNABLEO_COMPLY |

5.3.1.28.7 查询保持时间 (getHoldingTime)

| | |
|---|---|
| 定义 | |
| <pre>void getHoldingTime (out HoldingTime_T holdingTime) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询 EMS 的性能保持时间 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | holdingTime: 性能保持时间 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_NE_COMM_LOSS |

5.3.1.28.8 激活性能越限告警上报 (enableTCA)

| | |
|--|--|
| 定义 | |
| <pre>void enableTCA (in PMTPSelectList_T pmTPSelectList, out PMTPSelectList_T failedTPSelectList) raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 激活性能越限告警上报 |
| 输入参数 | pmTPSelectList: 性能选择参数列表 |
| 输入/输出参数 | 无 |
| 输出参数 | failedTPSelectList: 失败的性能选择参数列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_NE_COMM_LOSS EXCPT_UNABLEO_COMPLY |

5.3.1.28.9 去激活性能越限告警上报 (disableTCA)

| | |
|---|--|
| 定义 | |
| <pre>void disableTCA (in PMTPSelectList_T pmTPSelectList, out PMTPSelectList_T failedTPSelectList) raises (globaldefs::ProcessingFailureException) ;</pre> | |

| 说明 | |
|---------|--|
| 功能描述 | 去激活性能越限告警上报 |
| 输入参数 | pmTPSelectList: 性能选择参数列表 |
| 输入/输出参数 | 无 |
| 输出参数 | failedTPSelectList: 失败的性能选择参数列表 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_UNABLEO_COMPLY EXCPT_NE_COMM_LOSS |

5.3.1.28.10 查询终端点的历史性能数据 (getTPHistoryPMDData)

| 定义 | |
|---|--|
| <pre>void getTPHistoryPMDData (in PMTPSelectList_T pmTPSelectList, in PMPParameterNameList_T pmParameters, in globaldefs::Time_T startTime, in globaldefs::Time_T endTime, in unsigned long how_many, out PMDataList_T pmDataList, out PMDataIterator_I pmIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询指定终端点、时间段的历史性能数据 |
| 输入参数 | pmTPSelectList: 性能选择参数列表。 pmParameters: 性能参数。 startTime: 开始时间。 endTime: 结束时间。 how_many: 迭代查询数据方式下首次查询的数目 |
| 输入/输出参数 | 无 |
| 输出参数 | pmDataList: 性能数据列表。 pmIt: 迭代查询接口 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_UNABLEO_COMPLY EXCPT_INVALID_INPUT EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_TOO_MANY_OPEN_ITERATORS |

5.3.1.28.11 查询全部性能监测点 (getAllPMPs)

| 定义 | |
|--|--|
| <pre>void getAllPMPs (in globaldefs::NamingAttributes_T tpOrMeName, in unsigned long how_many, out PMPList_T pmpList, out PMPIterator_I pmpIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 指定对象名称, 查询全部性能监测点 |
| 输入参数 | tpOrMeName: 指定的终端点或网元名称。 how_many: 迭代查询数据方式下首次查询的数目 |
| 输入/输出参数 | 无 |
| 输出参数 | pmpList: 性能监测点列表。 pmpIt: 迭代查询性能监测点接口 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLEO_COMPLY EXCPT_NE_COMM_LOSS |

5.3.1.28.12 查询全部性能监测点名称 (getAllPMPNames)

| 定义 | |
|--|--|
| <pre>void getAllPMPNames (in globaldefs::NamingAttributes_T tpOrMeName, in unsigned long how_many, out globaldefs::NamingAttributesList_T nameList, out globaldefs::NamingAttributesIterator_I nameIt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 指定对象名称, 查询全部性能监测点名 |
| 输入参数 | tpOrMeName: 指定的终端点或网元名称。 how_many: 迭代查询数据方式下首次查询的数目 |
| 输入/输出参数 | 无 |
| 输出参数 | nameList: 顶层拓扑连接的名称列表。 nameIt: 迭代查询名称接口 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLEO_COMPLY EXCPT_NE_COMM_LOSS |

5.4 通用管理模块

5.4.1 会话管理模块 (module session)

会话部分具体实现有两个会话接口，一个实现在EMS，即EMSSession_I；另一个实现在NMS，即NMSSession_I，它们都是本接口的派生接口。

5.4.1.1 Session_I 接口

5.4.1.1.1 通信监测操作 (ping)

| 定义 | |
|---------------|---------|
| void ping (); | |
| 说明 | |
| 功能描述 | 探测通信的丢失 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | 无 |

5.4.1.1.2 结束会话 (endSession)

| 定义 | |
|----------------------------|---------------------------|
| oneway void endSession (); | |
| 说明 | |
| 功能描述 | 用于结束会话，NMS 和 EMS 都可以发起此操作 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | 无 |

5.4.2 EMS 会话厂管理模块 (module emsSessionFactory)

5.4.2.1 EmsSessionFactory_I 接口 (从 mtnmVersion::Version_I 继承)

EMS会话厂接口。提供了访问EMS 的入口。通过EMS会话厂可以进一步查询EMS会话接口。

5.4.2.1.1 查询 EMS 会话接口 (getEmsSession)

| 定义 | |
|---|--|
| <pre>void getEmsSession (in string user, in string password, in nmsSession::NmsSession_I client, out emsSession::EmsSession_I emsSessionInterface) raises (globaldefs::ProcessingFailureException); };</pre> | |
| 说明 | |
| 功能描述 | 操作允许NMS 查询 EmsSession_I 接口对象，其包含了 EMS 能够包含的管理者对象 |
| 输入参数 | <p>user: 用户名。</p> <p>password: 用户口令。</p> <p>client: NMS 的会话操作接口，用于 EMS 操作使用，如上报事件等</p> |

| | |
|---------|--|
| 输入/输出参数 | 无 |
| 输出参数 | emsSessionInterface: EMS 会话操作接口 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ACCESS_DENIED |

5.4.3 EMS 会话管理模块 (module emsSession)

5.4.3.1 EmsSession_I 接口 (从 session::Session_I 继承)

EMS会话管理接口, 用于提供对EMS访问操作的各管理者

5.4.3.1.1 管理者名称信息 (managerNames_T)

| | |
|---|----------------------------------|
| 定义 | |
| <code>typedef sequence<string> managerNames_T;</code> | |
| 说明 | |
| 对象说明 | 管理者名称信息。在 NMS 查询 EMS 支持的管理者信息时使用 |

5.4.3.1.2 查询 EMS 支持的管理者信息 (getSupportedManagers)

| | |
|--|--|
| 定义 | |
| <code>void getSupportedManagers (out managerNames_T supportedManagerList) raises (globaldefs::ProcessingFailureException);</code> | |
| 说明 | |
| 功能描述 | 查询 EMS 支持的管理者信息。根据查询到的管理者信息, 可以进一步查询管理者的操作接口 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | supportedManagerList: 管理者名称列表。要求必须一致的管理者名称有: "EMS": EMS 管理者。 "ManagedElement": 网元管理者。 "MultiLayerSubnetwork": 子网管理者 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_ACCESS_DENIED |

5.4.3.1.3 查询管理者操作接口 (getManager)

| | |
|---|--|
| 定义 | |
| <code>void getManager (in string managerName, out common::Common_I managerInterface) raises (globaldefs::ProcessingFailureException);</code> | |
| 说明 | |
| 功能描述 | 查询 EMS 支持的管理者信息。根据查询到的管理者信息, 可以进一步查询管理者的操作接口 |
| 输入参数 | managerName: 管理者名称 |
| 输入/输出参数 | 无 |
| 输出参数 | managerInterface: 管理者操作接口, 管理者的操作接口都是从 common 接口继承的 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_ACCESS_DENIED |

5.4.3.1.4 查询事件通道 (getEventChannel)

| 定义 | |
|--|---|
| <pre>void getEventChannel (out CosNotifyChannelAdmin::EventChannel eventChannel) raises (globaldefs::ProcessingFailureException) ; };</pre> | |
| 说明 | |
| 功能描述 | 查询 EMS 提供的事件通道。NMS 可以利用此事件通道接收 EMS 发送过来的事件 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | eventChannel: EMS 提供的事件通道 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_ACCESS_DENIED |

5.4.4 NMS 会话管理模块 (module nmsSession)

5.4.4.1 NmsSession_I 接口 (从 session::Session_I 继承)

NMS 会话管理接口。本接口提供给 EMS 操作使用，在查询 EMS 会话接口时提供。

5.4.4.1.1 通知事件丢失 (eventLossOccurred)

| 定义 | |
|--|---|
| <pre>void eventLossOccurred (in globaldefs::Time_T startTime, in string notificationId) ;</pre> | |
| 说明 | |
| 功能描述 | 事件丢失通知。当 EMS 向 NMS 推事件失败时，它会使用此操作向 NMS 发出事件丢失通知 |
| 输入参数 | startTime: 事件丢失的起始时间。 notificationId: 通知标识符 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | 无 |

5.4.4.1.2 通知事件丢失清除 (eventLossCleared)

| 定义 | |
|---|--|
| <pre>void eventLossCleared (in globaldefs::Time_T endTime) ;</pre> | |
| 说明 | |
| 功能描述 | 事件丢失清除通知。当 EMS 向 NMS 推事件由失败恢复正常时，它会使用此操作向 NMS 发出事件丢失清除通知 |
| 输入参数 | endTime: 事件丢失的结束时间 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | 无 |

5.4.5 传送参数管理模块 (module transmissionParameters)

5.4.5.1 层速率 (LayerRate_T)

| | |
|---|-----|
| 定义 | |
| <code>typedef short LayerRate_T;</code> | |
| 说明 | |
| 对象说明 | 层速率 |

5.4.5.2 层速率列表 (LayerRateList_T)

| | |
|---|-------|
| 定义 | |
| <code>typedef sequence<LayerRate_T> LayerRateList_T;</code> | |
| 说明 | |
| 对象说明 | 层速率列表 |

5.4.5.3 层速率及其相关的传送参数列表 (LayeredParameters_T)

| | |
|--|----------------|
| 定义 | |
| <pre>struct LayeredParameters_T { LayerRate_T layer; globaldefs::NVList_T transmissionParams; };</pre> | |
| 说明 | |
| 对象说明 | 层速率及其相关的传送参数列表 |
| 属性名 | 属性说明 |
| layer | 层速率 |
| transmissionParams | 层参数列表 |

5.4.5.4 层速率及参数列表的列表 (LayeredParameterList_T)

| | |
|--|-------------|
| 定义 | |
| <code>typedef sequence<LayeredParameters_T> LayeredParameterList_T;</code> | |
| 说明 | |
| 对象说明 | 层速率及参数列表的列表 |

5.4.6 时间同步管理模块 (module timeMgr)

5.4.6.1 EMSTimeMgr_I 接口 (从 common::Common_I 继承)

时间管理接口。主要提供对EMS时间的操作，如果EMS支持NTP协议，设置时间操作可选。

5.4.6.1.1 查询 EMS 时间 (getEMSTime)

| | |
|---|----------------------|
| 定义 | |
| <pre>void getEMSTime (out globaldefs::Time_T emsTime) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 用于 NMS 查询 EMS 时间 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | emsTime: EMS的当前时间 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.4.6.1.2 设置 EMS 时间 (setEMSTime) (可选)

| 定义 | |
|--|---------------------------------------|
| <pre>void setEMSTime (in globaldefs::Time_T setTime) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 用于 NMS 设置 EMS 时间, 在 EMS 不支持 NTP 协议时使用 |
| 输入参数 | setTime: 设置时间 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.4.7 维护管理模块 (module maintenanceOps)

5.4.7.1 维护操作 (MaintenanceOperation_T)

| 定义 | |
|---|---|
| <pre>typedef string MaintenanceOperation_T;</pre> | |
| 说明 | |
| 对象说明 | 维护操作类型, 具体定义如下: "FACILITY_LOOPBACK": 表示向设备侧环回。 "TERMINAL_LOOPBACK": 表示向终端侧环回。 "FACILITY_FORCED_AIS": 表示向设备侧强制插入告警指示信号。 "TERMINAL_FORCED_AIS": 表示向终端侧强制插入告警指示信号。 "FORCE_RDI": 表示强插入远端缺陷指示 |

5.4.7.2 维护操作模式 (MaintenanceOperationMode_T)

| 定义 | |
|--|--------|
| <pre>enum MaintenanceOperationMode_T { MOM_OPERATE, MOM_RELEASE };</pre> | |
| 说明 | |
| 对象说明 | 维护操作模式 |
| 类型取值 | 取值说明 |
| MOM_OPERATE | 执行维护操作 |
| MOM_RELEASE | 解除维护操作 |

5.4.7.3 当前维护操作 (CurrentMaintenanceOperation_T)

| 定义 | |
|---|--|
| <pre>struct CurrentMaintenanceOperation_T { globaldefs::NamingAttributes_T tpName; MaintenanceOperation_T maintenanceOperation; transmissionParameters::LayerRate_T layerRate; globaldefs::NVSList_T additionalInfo; };</pre> | |

| 说明 | |
|----------------------|---------|
| 对象说明 | 当前的维护操作 |
| 属性名 | 属性说明 |
| tpName | 终端点名称 |
| maintenanceOperation | 维护操作类型 |
| layerRate | 层速率 |
| additionalInfo | 附加信息 |

5.4.7.4 当前维护操作列表 (CurrentMaintenanceOperationList_T)

| 定义 | |
|--|-----------|
| typedef sequence<CurrentMaintenanceOperation_T> CurrentMaintenanceOperationList_T; | |
| 说明 | |
| 对象说明 | 当前维护操作列表。 |

5.4.7.5 CurrentMaintenanceOperationIterator_I 接口

5.4.7.5.1 从迭代器中查询数据 (next_n)

| 定义 | |
|--|--|
| boolean next_n (in unsigned long how_many, out CurrentMaintenanceOperationList_T cMOList) raises (globaldefs::ProcessingFailureException); | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时，系统在数据取完后应该自动销毁该迭代器对象。 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目。 |
| 输入/输出参数 | 无。 |
| 输出参数 | cMOList: 首次查询返回的维护操作数据列表。 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.4.7.5.2 查询迭代器数据记录条数 (getLength)

| 定义 | |
|--|---|
| unsigned long getLength () raises (globaldefs::ProcessingFailureException); | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数 (注意这里指的是迭代器中数据记录的总条数, 该值在迭代器生命周期内是不变的) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.4.7.5.3 销毁迭代器对象 (destroy)

| 定义 | |
|---|--|
| void destroy () raises (globaldefs::ProcessingFailureException); | |

| 说明 | |
|---------|----------------------|
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.4.7.6 MaintenanceMgr_I 接口（从 common::Common_I 继承）

维护操作管理接口。主要提供执行维护操作和获取当前的维护操作数据等操作。

5.4.7.6.1 执行或解除维护操作（performMaintenanceOperation）

| 定义 | |
|---|--|
| <pre>void performMaintenanceOperation (in CurrentMaintenanceOperation_T maintenanceOperation, in MaintenanceOperationMode_T maintenanceOperationMode) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 用于 NMS 向 EMS 下发维护操作命令，可以是执行维护操作或解除已经执行的维护操作 |
| 输入参数 | maintenanceOperation: 当前的维护操作信息。 maintenanceOperationMode: 维护操作的方式，包括执行或解除两种 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_NOT_IMPLEMENTED EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_UNABLE_TO_COMPLY EXCPT_NOT_IN_VALID_STATE |

5.4.7.6.2 查询当前的维护操作（getActiveMaintenanceOperations）

| 定义 | |
|---|--|
| <pre>void getActiveMaintenanceOperations (in globaldefs::NamingAttributes_T tpOrMeName, in unsigned long how_many, out CurrentMaintenanceOperationList_T currentMaintenanceOpeationList, out CurrentMaintenanceOperationIterator_I cmolt) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 用于 NMS 查询网元或终端点上当前的维护操作信息 |
| 输入参数 | tpOrMeName: 终端点或网元名称。 how_many: 迭代查询数据时首次查询的数目 |
| 输入/输出参数 | 无 |
| 输出参数 | currentMaintenanceOpeationList: 当前的维护操作列表。 cmolt: 迭代查询当前的维护操作接口 |

| | |
|------|---|
| 操作异常 | EXCPT_INTERNAL_ERROR EXCPT_NOT_IMPLEMENTED EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_NE_COMM_LOSS EXCPT_UNABLE_TO_COMPLY EXCPT_TOO_MANY_OPEN_ITERATORS |
|------|---|

5.5 公共管理模块

5.5.1 全局定义 (module globaldefs)

5.5.1.1 名值对 (NameAndStringValue_T)

| | |
|--|---|
| 定义 | |
| <pre>struct NameAndStringValue_T { string name; string value; };</pre> | |
| 说明 | |
| 对象说明 | NameAndStringValue_T 结构用于替代 OMG 定义的名值对列表 (NVList)，从性能和开销的角度考虑，将 value 定义为 string 比将其定义为 any 类型更有效，本接口使用元组序列来表示对象的命名 |
| 属性名 | 属性说明 |
| name | 对象的名称 |
| value | 对象的值 |

5.5.1.2 名值对列表 (NVSList_T)

| | |
|---|-------|
| 定义 | |
| typedef sequence<NameAndStringValue_T> NVSList_T; | |
| 说明 | |
| 对象说明 | 名值对列表 |

5.5.1.3 对象的名值对 (NamingAttributes_T)

| | |
|---------------------------------------|--------|
| 定义 | |
| typedef NVSList_T NamingAttributes_T; | |
| 说明 | |
| 对象说明 | 对象的名值对 |

5.5.1.4 对象的名值对列表 (NamingAttributesList_T)

| | |
|--|----------|
| 定义 | |
| typedef sequence<NamingAttributes_T> NamingAttributesList_T; | |
| 说明 | |
| 对象说明 | 对象的名值对列表 |

5.5.1.5 时间 (Time_T)

| | |
|------------------------|--|
| 定义 | |
| typedef string Time_T; | |
| 说明 | |

| | |
|------|--|
| 对象说明 | <p>时间。其表示方式遵循 ITU-T X.208 中关于时间表示的定义，其格式为 "yyyyMMddhhmmss.s[Z {+ -}HHMm]"。其中：</p> <p>yyyy: 年，取值为"0000"…"9999"；</p> <p>MM: 月，取值为"01"…"12"；</p> <p>dd: 日，取值为"01"…"31"；</p> <p>hh: 时，取值为"00"…"23"；</p> <p>mm: 分，取值为"00"…"59"；</p> <p>ss: 秒，取值为"00"…"59"；</p> <p>Z: 表示 UTC，不是本地时间，取值"Z"；</p> <p>{+ -}: 与 UTC 时间的时差，取值"+"或"-";</p> <p>HH: 时差中的小时部分，取值"00"…"23"；</p> <p>Mm: 时差中的分钟部分，取值"00"…"59"</p> |
|------|--|

5.5.1.6 连接方向 (ConnectionDirection_T)

| | |
|--|---------------------------|
| 定义 | |
| <pre>enum ConnectionDirection_T { CD_UNI, CD_BI };</pre> | |
| 说明 | |
| 对象说明 | 连接方向。可表示子网连接、交叉连接、拓扑连接的方向 |
| 属性名 | 属性说明 |
| CD_UNI | 单向 |
| CD_BI | 双向 |

5.5.1.7 异常类型 (ExceptionType_T)

| | |
|---|--|
| 定义 | |
| <pre>enum ExceptionType_T { EXCPT_NOT_IMPLEMENTED, EXCPT_INTERNAL_ERROR, EXCPT_INVALID_INPUT, EXCPT_OBJECT_IN_USE, EXCPT_TP_INVALID_ENDPOINT, EXCPT_ENTITY_NOT_FOUND, EXCPT_TIMESLOT_IN_USE, EXCPT_PROTECTION_EFFORT_NOT_MET, EXCPT_NOT_IN_VALID_STATE, EXCPT_UNABLE_TO_COMPLY, EXCPT_NE_COMM_LOSS, EXCPT_CAPACITY_EXCEEDED, EXCPT_ACCESS_DENIED, EXCPT_TOO_MANY_OPEN_ITERATORS, EXCPT_UNSUPPORTED_ROUTING_CONSTRAINTS, EXCPT_USERLABEL_IN_USE };</pre> | |

| 说明 | |
|---------------------------------------|---|
| 对象说明 | 在处理出错时抛出的异常类型 |
| 属性名 | 属性说明 |
| EXCPT_NOT_IMPLEMENTED | 操作没有完成或不支持操作 |
| EXCPT_INTERNAL_ERROR | EMS 内部处理错误 |
| EXCPT_INVALID_INPUT | 无效的输入 |
| EXCPT_OBJECT_IN_USE | 对象正被使用中。如配置的终端点已被用作交叉连接或被终结或已被映射，就不能再被用来做有冲突的配置 |
| EXCPT_TP_INVALID_ENDPOINT | 指定的终端点不存在或不能被创建 |
| EXCPT_ENTITY_NOT_FOUND | 实体没有找到。如使用 EMS 支持的某个对象名称作为参数来操作某个对象，但又没有找到此对象，就可抛出此异常 |
| EXCPT_TIMESLOT_IN_USE | 创建和激活子网连接时涉及的时隙正在被 EMS 所占用 |
| EXCPT_PROTECTION_EFFORT_NOT_MET | NMS 要求 SNC 的保护尽力程度 EMS 不能满足 |
| EXCPT_NOT_IN_VALID_STATE | 当前的状态不能进行某一操作。如不能删除处于激活状态的 SNC |
| EXCPT_UNABLE_TO_COMPLY | 在 EMS 不能响应操作请求时抛出 |
| EXCPT_NE_COMM_LOSS | 与网元的通信中断 |
| EXCPT_CAPACITY_EXCEEDED | 当某一操作请求超出了 EMS 或网元的处理能力时抛出 |
| EXCPT_ACCESS_DENIED | 当操作的安全认证失败时抛出 |
| EXCPT_TOO_MANY_OPEN_ITERATORS | 当请求迭代操作时总的迭代器数目超出了 EMS 的迭代器数目限制时抛出 |
| EXCPT_UNSUPPORTED_ROUTING_CONSTRAINTS | EMS 不支持设置的路由限制 |
| EXCPT_USERLABEL_IN_USE | 用户友好名称正在使用中，即友好名称已存在 |

5.5.1.8 处理失败时的异常类型 (ProcessingFailureException)

| 定义 | |
|---|---|
| <pre>exception ProcessingFailureException { ExceptionType_T exceptionType; string errorReason; };</pre> | |
| 说明 | |
| 对象说明 | 处理失败抛出的异常 |
| 属性名 | 属性说明 |
| exceptionType | 异常类型 |
| errorReason | 产生异常的原因。是对前面的 ExceptionType 进行补充说明，对 errorReason 的具体格式没有定义，只要是一个说明出错原因的可读 string 即可 |

5.5.1.9 NamingAttributesIterator_I 接口

5.5.1.9.1 从迭代器中查询数据 (next_n)

| 定义 | |
|--|--|
| <pre>boolean next_n (in unsigned long how_many, out NamingAttributesList_T nameList) raises (globaldefs::ProcessingFailureException);</pre> | |

| 说明 | |
|---------|---|
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时，系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many : 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | $nameList$: 首次查询返回的名值对列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.5.1.9.2 查询迭代器数据记录条数 (getLength)

| 定义 | |
|--|---|
| <pre>unsigned long getLength () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数（注意这里指的是迭代器中数据记录的总条数，该值在迭代器生命周期内是不变的） |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.5.1.9.3 销毁迭代器对象 (destroy)

| 定义 | |
|---|----------------------|
| <pre>void destroy () raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.5.2 通用接口模块 (module common)

本接口是管理对象的通用操作接口，其他管理对象类接口模块都要从本接口派生。

5.5.2.1 对象的任意能力 (Capability_T)

| 定义 | |
|---|---------|
| <pre>typedef globaldefs::NameAndStringValue_T Capability_T;</pre> | |
| 说明 | |
| 对象说明 | 对象的任意能力 |

5.5.2.2 对象的任意能力表述列表 (CapabilityList_T)

| 定义 | |
|---|-------------|
| <pre>typedef sequence<Capability_T> CapabilityList_T;</pre> | |
| 说明 | |
| 对象说明 | 对象的任意能力表述列表 |

5.5.2.3 Common_I 接口

5.5.2.3.1 设置对象网管本地名 (setNativeEMSName)

| 定义 | |
|--|--|
| <pre>void setNativeEMSName (in globaldefs::NamingAttributes_T objectName, in string nativeEMSName) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 指定对象名称, 设置该对象的本地名称 |
| 输入参数 | objectName: 对象名称。 nativeEMSName: 分配给对象的新的 EMS 名称 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLEO_COMPLY EXCPT_NE_COMM_LOSS |

5.5.2.3.2 设置对象所有者 (setOwner)

| 定义 | |
|--|--|
| <pre>void setOwner (in globaldefs::NamingAttributes_T objectName, in string owner) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 指定对象名称, 设置该对象所有者名称 |
| 输入参数 | objectName: 对象名称。 owner: 分配给对象的新的所有者名称 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLEO_COMPLY EXCPT_NE_COMM_LOSS |

5.5.2.3.3 设置对象用户标签 (setUserLabel)

| 定义 | |
|--|--|
| <pre>void setUserLabel (in globaldefs::NamingAttributes_T objectName, in string userLabel, in boolean enforceUniqueness) raises (globaldefs::ProcessingFailureException);</pre> | |

| 说明 | |
|---------|---|
| 功能描述 | 指定对象名称, 设置该对象用户标签属性 |
| 输入参数 | objectName: 对象名称。 userLabel: 分配给对象的新的用户标签。 enforceUniqueness: 用户标签是否唯一 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_ENTITY_NOT_FOUND EXCPT_UNABLO_COMPLY EXCPT_NE_COMM_LOSS |

5.5.2.3.4 设置对象附加属性 (setAdditionalInfo)

| 定义 | |
|---|--|
| <pre>void setAdditionalInfo (in globaldefs::NamingAttributes_T objectName, inout globaldefs::NVSLList_T additionalInfo) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 设置对象附加信息 |
| 输入参数 | objectName: 对象名称 |
| 输入/输出参数 | additionalInfo: 对象附加信息 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_NE_COMM_LOSS EXCPT_UNABLEO_COMPLY |

5.5.2.3.5 查询对象能力描述 (getCapabilities)

| 定义 | |
|---|---------------------------------------|
| <pre>void getCapabilities (out CapabilityList_T capabilitiesList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 查询对象能力, 例如是否支持交叉共享等 (增加环回能力度) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | capabilitiesList: 对象的任意能力表述列表, 类型为名值对 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.5.3 通知管理模块 (module notifications)

通知管理功能包括通知订购功能, 通知上报功能和事件同步功能。本部分通知格式定义(心跳通知、告警通知除外)请见TMF 814 (2003)中的“OMGServicesUsage.pdf”; CosNotification::StructuredEvent的定义见“CosNotification.idl”。

心跳通知的事件头、事件体格式均符合TMF 814 (2003)的定义, 其中可过滤体部分定义如下。

| 名称 | 类型 | 取值描述 |
|------------------|-----------------------------|-------------------------------|
| "notificationId" | string | 通知标识符, 用来在需要时唯一标识通知, 可进行通知的关联 |
| "object Name" | string | 发送通知的对象名称 |
| "objectType" | notifications::ObjectType_T | 发送通知的对象类型 |
| "emsTime" | globaldefs::Time_T | 发送心跳通知的时间 |

告警通知的事件头、事件体格式均符合TMF 814 (2003)的定义, 其中可过滤体部分定义如下。

| 名称 | 类型 | 取值描述 |
|---------------------------|--------------------------------------|---|
| "notificationId" | string | 通知标识符, 用来在需要时唯一标识通知, 可进行通知的关联 |
| "objectName" | globaldefs::NamingAttributes_T | 发送通知的对象名称。 |
| "nativeEMSName" | string | 发送通知的对象的本地EMS名称 |
| "nativeProbableCause" | string | 本地可能的原因 |
| "objectType" | notifications::ObjectType_T | 发送通知的对象类型 |
| "objectTypeQualifier" | notifications::ObjectTypeQualifier_T | 对象类型指示符, 标识对象是否为新创建的对象类型 |
| "emsTime" | globaldefs::Time_T | 告警开始的EMS时间 |
| "neTime" | globaldefs::Time_T | 告警开始的网元时间 |
| "emsEndTime" | globaldefs::Time_T | 告警结束的EMS时间, 可选 |
| "neEndTime" | globaldefs::Time_T | 告警结束的网元时间, 可选 |
| "isClearable" | boolean | 是否是可清除的事件 |
| "layerRate" | transmissionParameters::LayerRate_T | 层速率 |
| "probableCause" | string | 可能原因。 |
| "probableCauseQualifier" | string | 告警原因定位标识(通过objectName, layerRate, 和probableCause唯一标识一个告警) |
| "perceivedSeverity" | notifications::PerceivedSeverity_T | 告警级别 |
| "serviceAffecting" | notifications::ServiceAffecting_T | 是否影响服务 |
| "affectedTPLList" | globaldefs::NamingAttributesList_T | 影响的TP列表, 可选 |
| "additionalText" | string | 附加文本 |
| "additionalInfo" | globaldefs::NVSList_T | 附加信息, 可选 |
| "X.733::EventType" | string | 告警的基本类型, 可选。参照ITU-T X.733的规定, 取值共有如下5类: "communicationsAlarm", "environmentalAlarm", "equipmentAlarm", "processingErrorAlarm", "qualityofServiceAlarm" |
| "X.733::SpecificProblems" | notifications::SpecificProblemList_T | 优化后的告警原因, 可选 |

| | | |
|----------------------------------|---|---|
| "X.733::BackedUpStatus" | string | 备份状态, 可选。取值为"BACKED_UP", "NOT_BACKED_UP"之一 |
| "X.733::BackUpObject" | globaldefs::NamingAttributes_T | 备份对象, 提供通知备份服务的对象。该参数为条件必选, 当"X.733::BackedUpStatus"取值为"BACKED_UP", 该参数有值 |
| "X.733::TrendIndication" | string | 变化趋势, 可选。取值为"MORE_SEVERE", "NO_CHANGE", "LESS_SEVERE"当中的一个 |
| "X.733::CorrelatedNotifications" | notifications::CorrelatedNotificationList_T | 关联通知列表, 可选 |
| "X.733::MonitoredAttributes" | notifications::NVList_T | 监视属性, 可选 |
| "X.733::ProposedRepairActions" | notifications::ProposedRepairActionList_T | 故障处理的建议操作, 可选 |
| "X.733::AdditionalInfo" | notifications::NVList_T | 附加信息, 可选 |
| "rcaiIndicator" | Boolean | RACI指示符, 取值为"TRUE"表示该告警为根原因告警, 取值为"FALSE"表示该告警为原始告警。缺省时默认取值为"FALSE" |
| "acknowledgeIndication" | notifications::AcknowledgeIndication_T | 告警确认状态 |

5.5.3.1 告警级别 (PerceivedSeverity_T)

| 定义 | |
|--|------|
| <pre>enum PerceivedSeverity_T { PS_INDETERMINATE, PS_CRITICAL, PS_MAJOR, PS_MINOR, PS_WARNING, PS_CLEARED };</pre> | |
| 说明 | |
| 对象说明 | 告警级别 |
| 类型取值 | 取值说明 |
| PS_INDETERMINATE | 未确认 |
| PS_CRITICAL | 严重告警 |
| PS_MAJOR | 主要告警 |
| PS_MINOR | 次要告警 |
| PS_WARNING | 警告 |
| PS_CLEARED | 已清除 |

5.5.3.2 告警级别列表 (PerceivedSeverityList_T)

| 定义 | |
|---|--------|
| <pre>typedef sequence<PerceivedSeverity_T> PerceivedSeverityList_T;</pre> | |
| 说明 | |
| 对象说明 | 告警级别列表 |

5.5.3.3 告警标识 (AlarmId_T)

| | |
|--|----------|
| 定义 | |
| <pre>struct AlarmId_T { globaldefs::NamingAttributes_T objectName; transmissionParameters::LayerRate_T layerRate; ProbableCauseList_T probableCause; string probableCauseQualifier; };</pre> | |
| 说明 | |
| 对象说明 | 告警标识 |
| 属性名 | 属性说明 |
| objectName | 对象名称 |
| layerRate | 速率层次 |
| probableCause | 告警可能原因 |
| probableCauseQualifier | 告警原因定位标识 |

5.5.3.4 告警/事件原因 (ProbableCause_T)

| | |
|--|---------|
| 定义 | |
| <pre>typedef string ProbableCause_T;</pre> | |
| 说明 | |
| 对象说明 | 告警/事件原因 |

5.5.3.5 告警/事件原因列表 (ProbableCauseList_T)

| | |
|--|-----------|
| 定义 | |
| <pre>typedef sequence < ProbableCause_T > ProbableCauseList_T;</pre> | |
| 说明 | |
| 对象说明 | 告警/事件原因列表 |

5.5.3.6 对象的名称和取值 (NameAndAnyValue_T)

| | |
|--|--|
| 定义 | |
| <pre>struct NameAndAnyValue_T { string name; any value; };</pre> | |
| 说明 | |
| 对象说明 | 对象的名称和取值，这里的对象取值是 any 类型，本结构主要用于事件通知的通知结构中 |
| 属性名 | 属性说明 |
| name | 对象名称 |
| value | 对象取值 |

5.5.3.7 名值对列表 (NVList_T)

| | |
|---|----------------|
| 定义 | |
| <pre>typedef sequence <NameAndAnyValue_T> NVList_T;</pre> | |
| 说明 | |
| 对象说明 | 一个或多个对象的名称和取值对 |

5.5.3.8 事件信息列表 (EventList_T)

| | |
|--|--------|
| 定义 | |
| typedef sequence <CosNotification::StructuredEvent> EventList_T; | |
| 说明 | |
| 对象说明 | 事件信息列表 |

5.5.3.9 对象类型 (ObjectType_T)

| | |
|---|-------|
| 定义 | |
| <pre>enum ObjectType_T { OT_EMS, OT_MANAGED_ELEMENT, OT_MULTILAYER_SUBNETWORK, OT_TOPOLOGICAL_LINK, OT_SUBNETWORK_CONNECTION, OT_PHYSICAL_TERMINATION_POINT, OT_CONNECTION_TERMINATION_POINT, OT_TERMINATION_POINT_POOL, OT_EQUIPMENT HOLDER, OT_EQUIPMENT, OT_PROTECTION_GROUP, OT_TRAFFIC_DESCRIPTOR, OT_AID };</pre> | |
| 说明 | |
| 对象说明 | 对象类型 |
| 类型取值 | 取值说明 |
| OT_EMS | EMS |
| OT_MANAGED_ELEMENT | 管理单元 |
| OT_MULTILAYER_SUBNETWORK | 子网 |
| OT_TOPOLOGICAL_LINK | 拓扑连接 |
| OT_SUBNETWORK_CONNECTION | 子网连接 |
| OT_PHYSICAL_TERMINATION_POINT | 物理终端点 |
| OT_CONNECTION_TERMINATION_POINT | 子网终端点 |
| OT_TERMINATION_POINT_POOL | 终端点池 |
| OT_EQUIPMENT HOLDER | 设备容器 |
| OT_EQUIPMENT | 设备 |
| OT_PROTECTION_GROUP | 保护组 |
| OT_TRAFFIC_DESCRIPTOR | 通信描述符 |
| OT_AID | 未定义对象 |

5.5.3.10 对象类型指示符 (ObjectTypeQualifier_T)

| | |
|---------------------------------------|--------------------------|
| 定义 | |
| typedef string ObjectTypeQualifier_T; | |
| 说明 | |
| 对象说明 | 对象类型指示符, 标识对象是否为新创建的对象类型 |

5.5.3.11 告警确认状态 (AcknowledgeIndication_T)

| | |
|--|--------|
| 定义 | |
| <pre>enum AcknowledgeIndication_T { AI_EVENT_ACKNOWLEDGED, AI_EVENT_UNACKNOWLEDGED, AI_NA };</pre> | |
| 说明 | |
| 对象说明 | 告警确认状态 |
| 类型取值 | 取值说明 |
| AI_EVENT_ACKNOWLEDGED | 确认 |
| AI_EVENT_UNACKNOWLEDGED | 未确认 |
| AI_NA | 未知 |

5.5.3.12 故障处理建议操作 (ProposedRepairAction_T)

| | |
|---|----------|
| 定义 | |
| <pre>typedef string ProposedRepairAction_T;</pre> | |
| 说明 | |
| 对象说明 | 故障处理建议操作 |

5.5.3.13 故障处理建议操作列表 (ProposedRepairActionList_T)

| | |
|---|------------|
| 定义 | |
| <pre>typedef sequence< ProposedRepairAction_T > ProposedRepairActionList_T;</pre> | |
| 说明 | |
| 对象说明 | 故障处理建议操作列表 |

5.5.3.14 告警原因优化 (SpecificProblem_T)

| | |
|--|--------|
| 定义 | |
| <pre>typedef string SpecificProblem_T;</pre> | |
| 说明 | |
| 对象说明 | 告警原因优化 |

5.5.3.15 告警原因优化列表 (SpecificProblemList_T)

| | |
|---|----------|
| c | |
| <pre>typedef sequence< SpecificProblem_T > SpecificProblemList_T;</pre> | |
| 说明 | |
| 对象说明 | 告警原因优化列表 |

5.5.3.16 是否影响服务 (ServiceAffecting_T)

| | |
|--|--|
| 定义 | |
| <pre>enum ServiceAffecting_T { SA_UNKNOWN, SA_SERVICE_AFFECTING, SA_NON_SERVICE_AFFECTING };</pre> | |

| 说明 | |
|--------------------------|-------|
| 对象说明 | 服务影响 |
| 类型取值 | 取值说明 |
| SA_UNKNOWN | 未知 |
| SA_SERVICE_AFFECTING | 影响服务 |
| SA_NON_SERVICE_AFFECTING | 不影响服务 |

5.5.3.17 通知 ID 的列表

| 定义 | |
|---|-----------|
| typedef sequence<string> NotifIDList_T; | |
| 说明 | |
| 对象说明 | 通知 ID 的列表 |

5.5.3.18 关联通知 (CorrelatedNotifications_T)

| 定义 | |
|---|-------------|
| <pre>struct CorrelatedNotifications_T { globaldefs::NamingAttributes_T source; NotifIDList_T notifIDs; };</pre> | |
| 说明 | |
| 对象说明 | 关联通知 |
| 属性名 | 属性说明 |
| source | 发出关联通知的对象名称 |
| notifIDs | 关联通知的 ID |

5.5.3.19 关联通知列表 (CorrelatedNotificationList_T)

| 定义 | |
|---|--------|
| typedef sequence<CorrelatedNotifications_T> CorrelatedNotificationList_T; | |
| 说明 | |
| 对象说明 | 关联通知列表 |

5.5.3.20 EventIterator_I 接口

5.5.3.20.1 从迭代器中查询事件信息 (next_n)

| 定义 | |
|--|---|
| <pre>boolean next_n (in unsigned long how_many, out EventList_T eventList) raises (globaldefs::ProcessingFailureException);</pre> | |
| 说明 | |
| 功能描述 | 从迭代器中查询 n 条记录。当 $how_many \geq$ 迭代器剩余记录数时，系统在数据取完后应该自动销毁该迭代器对象 |
| 输入参数 | how_many: 首次迭代查询返回的数据数目 |
| 输入/输出参数 | 无 |
| 输出参数 | eventList: 首次查询返回的事件数据列表 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.5.3.2.2 查询迭代器记录条数 (getLength)

| 定义 | |
|---|---|
| <pre>unsigned long getLength () raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 查询迭代器中数据记录的总条数(注意这里指的是迭代器中数据记录的总条数,该值在迭代器生命周期内是不变的) |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

5.5.3.2.3 销毁迭代器对象 (destroy)

| 定义 | |
|--|----------------------|
| <pre>void destroy () raises (globaldefs::ProcessingFailureException) ;</pre> | |
| 说明 | |
| 功能描述 | 销毁迭代器对象 |
| 输入参数 | 无 |
| 输入/输出参数 | 无 |
| 输出参数 | 无 |
| 操作异常 | EXCPT_INTERNAL_ERROR |

附 录 A
(规范性附录)
模块与 IDL 的映射

本部分各模块与IDL文件间的映射关系如表A.1所示。

表A.1 模块与 IDL 的映射表

| 模块名称 | IDL 名称 | 包含 IDL 名称 |
|-----------------------|---------------------------|---|
| aSAP | aSAP.idl | globaldefs.idl notifications.idl |
| common | common.idl | globaldefs.idl |
| emsMgr | emsMgr.idl | common.idl multiLayerSubnetwork.idl notifications.idl aSAP.idl |
| emsSession | emsSession.idl | CosNotifyChannelAdmin.idl common.idl session.idl |
| emsSessionFactory | emsSessionFactory.idl | globaldefs.idl session.idl emsSession.idl nmsSession.idl mtnmVersion.idl |
| equipment | equipment.idl | common.idl globaldefs.idl terminationPoint.idl |
| globaldefs | globaldefs.idl | — |
| maintenanceOps | maintenanceOps.idl | common.idl transmissionParameters.idl |
| managedElement | managedElement.idl | transmissionParameters.idl |
| managedElementManager | managedElementManager.idl | globaldefs.idl common.idl managedElement.idl transmissionParameters.idl terminationPoint.idl notifications.idl subnetworkConnection.idl aSAP.idl |
| mtnmVersion | mtnmVersion.idl | — |
| multiLayerSubnetwork | multiLayerSubnetwork.idl | globaldefs.idl common.idl subnetworkConnection.idl transmissionParameters.idl managedElement.idl topologicalLink.idl terminationPoint.idl |

表 A.1 (续)

| 模块名称 | IDL 名称 | 包含 IDL 名称 |
|------------------------|----------------------------|--|
| nmsSession | nmsSession.idl | session.idl globaldefs.idl |
| notifications | notifications.idl | globaldefs.idl CosNotification.idl performance.idl transmissionParameters.idl |
| performance | performance.idl | transmissionParameters.idl common.idl |
| protection | protection.idl | globaldefs.idl transmissionParameters.idl common.idl |
| session | session.idl | |
| subnetworkConnection | subnetworkConnection.idl | globaldefs.idl transmissionParameters.idl terminationPoint.idl |
| terminationPoint | terminationPoint.idl | globaldefs.idl transmissionParameters.idl |
| timeMgr | timeMgr.idl | globaldefs.idl common.idl |
| topologicalLink | topologicalLink.idl | globaldefs.idl transmissionParameters.idl |
| transmissionParameters | transmissionParameters.idl | globaldefs.idl |

广东省网络空间安全协会受控资料

中华人民共和国
通信行业标准
光传送网（OTN）网络管理技术要求
第5部分：基于IDL/IIOP技术的EMS-NMS接口信息模型

YD/T 2149.5-2011

*

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
宝隆元（北京）印刷技术有限公司印刷

*

开本：880×1230 1/16 2012年1月第1版
印张：7 2012年1月北京第1次印刷
字数：196千字

ISBN 978 - 7 - 115 - 2495/ 12 - 73

定价：70元