目 录

1 系统概述.....1-1
1.1 引言.....1-2
1.2 遵从的标准介绍.....1-2
1.3 特性.....1-2
1.4 在综合网管中的地位.....1-3
2 系统特点.....2-1
2.1 新增功能.....2-2
2.2 功能特色.....2-2
2.3 技术指标.....2-2
3 系统功能.....3-1
3.1 功能概述.....3-2
3.2 EMS 管理.....3-2
3.3 网元管理.....3-3
3.4 设备管理.....3-6
3.5 故障管理.....3-7
3.6 性能管理.....3-8
3.7 保护管理.....3-10
3.8 维护命令.....3-12
3.9 拓扑管理.....3-13
3.10 子网连接管理.....3-13
3.11 MSTP 管理.....3-18
3.11.1 MSTP 存量管理.....3-18
3.11.2 MSTP 业务管理.....3-25
3.11.3 MSTP 保护管理.....3-26
3.11.4 流量描述符管理.....3-28
3.11.5 封装层链路管理.....3-29
3.11.6 流域管理.....3-30
3.12 控制平面管理.....3-32
3.13 界面直通.....3-33
3.14 通知上报.....3-34

3.15 安全管理 3-38
3.16 异常管理 3-38
4 系统配置与组网应用 4-1
4.1 硬件配置 4-2
4.2 软件配置 4-2
4.3 组网方式 4-2
4.4 高可用性系统配置 4-3
4.4.1 Watchman 高可用性系统 4-3
4.4.2 Veritas 温备份高可用性系统 4-6
4.4.3 Veritas 热备份高可用性系统 4-7
4.4.4 Sun Cluster 高可用性系统 4-10
5 系统安装和运行 5-1
5.1 检查与准备工作 5-2
5.1.1 检查 T2000 License 5-2
5.1.2 检查 T2000 配置工具 5-2
5.1.3 检查通知服务 5-3
5.2 启动和禁用 CORBA 接口(单机) 5-4
5.2.1 启动 T2000 CORBA 接口 5-5
5.2.2 重新启动 T2000 CORBA 接口 5-14
5.2.3 禁用 T2000 CORBA 接口 5-15
5.3 启动和禁用 CORBA 接口(双机) 5-17
5.3.1 Solaris 10 系统 5-17
5.3.2 Solaris 8 系统 5-31
5.4 附录 5-42
5.4.1 License 项与 CORBA 的关系 5-42
6 维护操作 6-1
6.1 维护条件说明 6-2
6.2 日常维护操作 6-2
6.3 常见问题处理 6-3
6.3.1 启动 T2000 CORBA 接口失败应该怎么处理 6-3
6.3.2 如何判断 T2000 已经成功打开 CORBA 接口 6-3
6.3.3 T2000 CORBA 接口和上层网管对接需要做哪些准备工作 6-3
6.3.4 如何判断 T2000 网管是否支持 CORBA 接口功能 6-4
7 接口信息模型 7-1
7.1 背景知识 7-2
7.1.1 管理器对象定义 7-2
7.1.2 实体对象定义 7-3
7.2 功能实现声明 7-5

7.2.1 说明 7-5
7.2.2 常用参数 7-6
7.2.3 Common 模块 7-7
7.2.4 EmsMgr 模块 7-8
7.2.5 Ems Session 模块 7-10
7.2.6 EmsSessionFactory 模块 7-10
7.2.7 Equipment 模块 7-11
7.2.8 GuiCutThrough 模块 7-16
7.2.9 MaintenanceOperations 模块 7-17
7.2.10 ManagedElement 模块 7-20
7.2.11 ManagedElementMgr 模块 7-21
7.2.12 MTNM Version 模块 7-25
7.2.13 MultiLayerSubnetwork 模块 7-26
7.2.14 Performance 模块 7-33
7.2.15 Protection 模块 7-36
7.2.16 Session 模块 7-42
7.2.17 Subnetwork Connection 模块 7-43
7.2.18 Termination Point 模块 7-46
7.2.19 Topological Link 模块 7-47
7.2.20 HW_mstpInventory 模块 7-48
7.2.21 HW_mstpService 模块 7-56
7.2.22 HW_mstpProtect 模块 7-61
7.2.23 trafficDescriptor 模块 7-66
7.2.24 控制平面管理模块 7-69
7.2.25 FlowDomain 模块 7-71
7.2.26 EncapsulationLayerLink 模块 7-76
7.2.27 TopoMgr 模块 7-78
7.3 CORBA 服务说明 7-80
7.3.1 概述 7-80
7.3.2 名字服务 7-80
7.3.3 通知服务 7-83
7.4 通知事件格式 7-86
7.4.1 NT_ALARM 事件格式 7-86
7.4.2 NT_TCA 事件格式 7-87
7.4.3 NT_ FILE_TRANSFER_STATUS 事件格式 7-89
7.4.4 NT_OBJECT_CREATION 事件格式 7-89
7.4.5 NT_OBJECT_DELETION 事件格式 7-91
7.4.6 NT_ATTRIBUTE_VALUE_CHANGE 事件格式 7-92
7.4.7 NT_STATE_CHANGE 事件格式 7-93

7.4.8 NT_PROTECTION_SWITCH 事件格式 7-93
7.4.9 NT_ATMPROTECTION_SWITCH 事件格式 7-95
7.4.10 NT_WDMPROTECTION_SWITCH 事件格式 7-95
7.4.11 NT_RPRPROTECTION_SWITCH 事件格式 7-96
7.4.12 NT_EPROTECTION_SWITCH 事件格式 7-97
7.4.13 NT_ROUTE_CHANGE 事件格式 7-98
7.4.14 NT_ASON_RESOURCE_CHANGE 事件格式 7-98
7.4.15 NT_PRBSTEST_STATUS 事件格式 7-99
7.4.16 NT_HEARTBEAT 事件格式 7-99
7.5 层速率说明 7-100
7.6 传输参数说明 7-103
7.7 AdditionalInfo 使用说明 7-108
7.7.1 ManagedElement_T 7-109
7.7.2 TerminationPoint_T 7-110
7.7.3 EMS_T 7-110
7.7.4 Equipment_T 7-111
7.7.5 EquipmentHolder_T 7-111
7.7.6 SubnetworkConnection_T 7-113
7.7.7 SNCCreateData_T 7-113
7.7.8 SNCModifyData_T 7-114
7.7.9 ProtectionSubnetwork_T 7-114
7.7.10 CrossConnect_T 7-114
7.7.11 HW_EthService_T 7-115
7.7.12 EthernetOAMOperation_T 7-115
7.7.13 ELLinkCreateData_T 7-115
7.7.14 FDFrCreateData_T 7-116
7.7.15 NT_ALARM 7-116
7.7.16 NT_TCA 7-117
7.7.17 NT_PROTECTION_SWITCH 7-117
7.7.18 NT_EPROTECTION_SWITCH 7-117
7.7.19 NT_WDMPROTECTION_SWITCH 7-118
7.8 接口限制和约束说明 7-118
7.8.1 约束与限制 1 7-118
7.8.2 约束与限制 2 7-118
7.8.3 约束与限制 3 7-118
7.8.4 约束与限制 4 7-118
7.8.5 约束与限制 5 7-119
7.8.6 约束与限制 6 7-120
7.8.7 约束与限制 7 7-120
7.8.8 约束与限制 8 7-120

7.8.9 约束与限制 9 7-120
7.9 对象命名规则 7-121
7.9.1 EMS 7-122
7.9.2 Subnetwork 7-122
7.9.3 TopoSubnetwork 7-122
7.9.4 ProtectionSubnetwork 7-123
7.9.5 SubnetworkConnection 7-123
7.9.6 ManagedElement 7-123
7.9.7 TopologicalLink 7-124
7.9.8 EPGP 7-124
7.9.9 PTP 7-124
7.9.10 CTP 7-125
7.9.11 TrafficDescriptor 7-127
7.9.12 EquipmentHolder 7-127
7.9.13 Equipment 7-128
7.9.14 ProtectionGroup 7-128
7.9.15 WDM ProtectionGroup 7-128
7.9.16 VirtualBridge 7-129
7.9.17 VLAN 7-129
7.9.18 Ethernet Service 7-129
7.9.19 ATM Service 7-130
7.9.20 ATM ProtectGroup 7-130
7.9.21 QoS Rule 7-130
7.9.22 Flow 7-131
7.9.23 Flow Domain 7-131
7.9.24 FlowDomainFragment 7-131
7.9.25 EncapsulationLayerLink 7-132
7.9.26 LinkAggregationGroup 7-132
7.9.27 RPRNode 7-132
7.9.28 Routing Area 7-133
7.9.29 SNPPLink 7-133
7.10 非功能互通性声明 7-133
7.10.1 EMS 子网配置 7-133
7.10.2 SNC 模式 7-133
7.10.3 迭代器使用 7-135
7.10.4 心跳检测机制 7-135
7.10.5 BT 命名格式使用说明 7-137
7.10.6 SSL 7-140

插图目录

图 1-1 T2000 CORBA 接口在综合网管中的地位 1-3
图 4-1 T2000 CORBA 接口在网管体系中的组网方式 4-2
图 4-2 T2000 CORBA 接口服务的集中部署方式 4-3
图 4-3 T2000 CORBA 接口服务的分开部署方式 4-3
图 4-4 硬件结构图 4-4
图 4-5 软件结构图 4-5
图 4-6 硬件结构图 4-6
图 4-7 软件结构图 4-7
图 4-8 硬件结构图 4-8
图 4-9 软件结构图 4-9
图 4-10 硬件连接图-高可用性系统（Sun Cluster） 4-10
图 4-11 软件结构图-高可用性系统（Sun Cluster） 4-11
图 7-1 TMF 建议的名字图 7-80
图 7-2 iManager T2000 名字图 7-81
图 7-3 ping 操作 7-136
图 7-4 心跳通知 7-136

表格目录

表 3-1 支持上报的属性 3-35
表 3-2 支持上报的状态 3-36
表 4-1 配置要求-高可用性系统（Watchman）服务器端 4-5
表 4-2 配置要求-高可用性系统（Veritas）服务器端 4-7
表 4-3 配置要求-高可用性系统（Veritas）服务器端 4-9
表 4-4 配置要求-高可用性系统（Sun Cluster）服务器端 4-11
表 4-5 配置要求-高可用性系统（Sun Cluster）控制台 4-12
表 5-1 CORBA 模块与 License 项的对应关系表 5-42
表 5-2 CORBA 高级功能模块与 T2000 License 支持项的对应关系表 5-43
表 7-1 管理对象定义与管理对象名字关系 7-2
表 7-2 实体对象定义关系 7-3
表 7-3 Common_I 接口描述 7-7
表 7-4 EMS_T 的数据类型描述 7-8
表 7-5 EMSMgr_I 模块的接口描述 7-9
表 7-6 EmsSession_I 模块的接口描述 7-10
表 7-7 EmsSessionFactory_I 接口描述 7-10
表 7-8 Equipment_T 的数据类型描述 7-11
表 7-9 EquipmentHolder_T 的数据类型描述 7-12
表 7-10 Shelf_T 的数据类型描述 7-12
表 7-11 Cabinet_T 的数据类型描述 7-13
表 7-12 EquipmentRoom_T 的数据类型描述 7-13
表 7-13 EquipmentInventoryMgr_I 的接口描述 7-14
表 7-14 GCTProfileInfo_T 的数据类型描述 7-16
表 7-15 GuiCutThroughData_T 的数据类型描述 7-16
表 7-16 GuiCutThrouthMgr_I 的接口描述 7-17

表 7-17 CurrentMaintenanceOperation_T 的类型描述.....7-17
表 7-18 PRBSTestParameter_T 的类型描述.....7-18
表 7-19 PRBSTestResult_T 的类型描述.....7-18
表 7-20 MaintenanceMgr_I 的接口描述.....7-18
表 7-21 Managed Element 模块的数据类型描述.....7-20
表 7-22 ManagedElementMgr_I 模块的接口描述.....7-21
表 7-23 Version_I 的接口描述.....7-26
表 7-24 MultiLayerSubnetwork_T 模块的数据类型描述.....7-26
表 7-25 MultiLayerSubnetworkMgr_I 的接口描述.....7-27
表 7-26 PMData_T 的数据类型描述.....7-33
表 7-27 PMMeasurement_T 的数据类型描述.....7-33
表 7-28 PMThresholdValue_T 的数据类型描述.....7-34
表 7-29 PerformanceManagementMgr_I 的接口描述.....7-34
表 7-30 ProtectionGroup_T 的数据类型描述.....7-37
表 7-31 ProtectionSubnetwork_T 的数据类型描述.....7-38
表 7-32 EprotectGroup_T 的数据类型描述.....7-38
表 7-33 WDMprotectGroup_T 的数据类型描述.....7-39
表 7-34 ProtectionMgr_I 的接口描述.....7-39
表 7-35 Session_I 模块的接口描述.....7-43
表 7-36 Crossconnection_T 的数据类型描述.....7-43
表 7-37 SubnetworkConnection_T 的数据类型描述.....7-44
表 7-38 TerminationPoint_T 据类型描述.....7-46
表 7-39 TopologicalLink_T 模块的数据类型描述.....7-47
表 7-40 HW_MSTPEndPoint_T 的数据类型描述.....7-48
表 7-41 HW_VirtualBridge_T 的数据类型描述.....7-48
表 7-42 HW_VirtualLAN_T 的数据类型描述.....7-49
表 7-43 HW_MSTPBindingPath_T 的数据类型描述.....7-49
表 7-44 HW_ForwardEndPoint_T 的数据类型描述.....7-50
表 7-45 HW_QosRule_T 的数据类型描述.....7-50
表 7-46 HW_Flow_T 的数据类型描述.....7-50
表 7-47 HW_LinkAggregationGroup_T 的数据类型描述.....7-51
表 7-48 HW_MSTPInventoryMgr_I 的接口描述.....7-52

表 7-49 HW_ETHServiceTP_T 的数据类型描述 7-56
表 7-50 HW_ETHServiceCreateData_T 的数据类型描述 7-57
表 7-51 HW_ETHService_T 的数据类型描述 7-57
表 7-52 HW_ATMServiceTP_T 的数据类型描述 7-58
表 7-53 HW_ATMService_T 的数据类型描述 7-58
表 7-54 HW_ATMServiceCreateData_T 的数据类型描述 7-59
表 7-55 HW_MSTPServiceMgr_I 的接口描述 7-59
表 7-56 HW_RPRNode_T 的数据类型描述 7-61
表 7-57 HW_RPRSwitchData_T 的数据类型描述 7-61
表 7-58 HW_RPRTopoInfo_T 的数据类型描述 7-62
表 7-59 HW_ATMServiceProtectPair_T 的数据类型描述 7-62
表 7-60 HW_ATMSingleEndSwitchPara_T 的数据类型描述 7-63
表 7-61 HW_ATMPGSwitchData_T 的数据类型描述 7-63
表 7-62 HW_ATMProtectGroup_T 的数据类型描述 7-63
表 7-63 HW_ATMPGSingEndPara_T 的数据类型描述 7-64
表 7-64 HW_MSTPProtectMgr_I 的接口描述 7-64
表 7-65 TrafficDescriptor_T 的数据类型描述 7-66
表 7-66 TDCreateData _T 的数据类型描述 7-67
表 7-67 TrafficDescriptorMgr_I 的接口描述 7-67
表 7-68 HW_SnppLink_T 的数据类型描述 7-69
表 7-69 HW_controlPlaneMgr_I 的接口描述 7-70
表 7-70 FlowDomain _T 的数据类型描述 7-71
表 7-71 FlowDomainFragment_T 的数据类型描述 7-72
表 7-72 FlowDomainMgr_I 的接口描述 7-72
表 7-73 EncapsulationLayerLink_T 的数据类型描述 7-76
表 7-74 EncapsulationLayerLinkMgr_I 的接口描述 7-77
表 7-75 Node_T 的数据类型描述 7-79
表 7-76 TopoMgr_I 的接口描述 7-79
表 7-77 通知服务支持表 7-83
表 7-78 NT_ALARM 事件格式 7-86
表 7-79 NT_TCA 事件格式 7-87
表 7-80 NT_ FILE_TRANSFER_STATUS 事件格式 7-89

表 7-81 NT_OBJECT_CREATION 事件格式 7-89
表 7-82 NT_OBJECT_CREATION 事件结构补充（remainder_of_body）_创建网元 7-90
表 7-83 NT_OBJECT_CREATION 事件结构补充（remainder_of_body）_创建单板 7-90
表 7-84 NT_OBJECT_DELETION 事件格式 7-91
表 7-85 NT_ATTRIBUTE_VALUE_CHANGE 事件格式 7-92
表 7-86 NT_STATE_CHANGE 事件格式 7-93
表 7-87 NT_PROTECTION_SWITCH 事件格式 7-93
表 7-88 NT_ATMPROTECTION_SWITCH 事件格式 7-95
表 7-89 NT_WDMPROTECTION_SWITCH 事件格式 7-95
表 7-90 NT_RPRPROTECTION_SWITCH 事件格式 7-96
表 7-91 NT_EPROTECTION_SWITCH 事件格式 7-97
表 7-92 NT_ROUTE_CHANGE 事件格式 7-98
表 7-93 NT_ASON_RESOURCE_CHANGE 事件格式 7-98
表 7-94 NT_PRBSTEST_STATUS 事件格式 7-99
表 7-95 NT_HEARTBEAT 事件格式 7-99
表 7-96 T2000 CORBA 接口支持的层速率列表 7-100
表 7-97 传输参数说明 7-103
表 7-98 SNC 状态模式 7-134
表 7-99 ManagedElement 对象 7-137
表 7-100 SubnetworkConnection 对象 7-138
表 7-101 EncapsulationLayerLink 对象 7-138
表 7-102 FlowDomainFragment 对象 7-139
表 7-103 TopologicalLink 对象 7-139
表 7-104 EquipmentHolder 对象 7-139

1 系统概述

关于本章

本章描述内容如下表所示。

标题	内容
1.1 引言	T2000 CORBA 接口总体概述。
1.2 遵从的标准介绍	T2000 CORBA 接口遵从的标准。
1.3 特性	T2000 CORBA 接口特性概述。
1.4 在综合网管中的地位	T2000 CORBA 接口在综合网管中的地位。

1.1 引言

iManager T2000 参考 TMF（Telecommunication Management Forum）、MTNM（Multi-Technology Network Management）系列建议中与北向 CORBA 接口相关的内容，结合自身特点开发出了 T2000 CORBA 接口。T2000 CORBA 接口是网元管理层和网络管理层之间的通信枢纽。通过采用 CORBA 这一广受业界推崇的分布式对象技术，使得 iManager T2000 能够很好地适应网管综合化、跨域化的发展趋势。

1.2 遵从的标准介绍

iManager T2000 遵循 TMF MTNM 系列标准，实现了 NML（Network Management Level）、EML（Element Management Level）之间的 CORBA IDL（Interface Definition Language）接口。通过该接口，上层综合网管系统或 OSS 系统可以实现对华为传送网络的管理。

T2000 CORBA 接口所遵从的 TMF 标准如下：

- TMF 513 MTNM Business Agreement
- TMF 608 MTNM Information Agreement
- TMF 814 MTNM CORBA Solution Set
- TMF 814A MTNM Implementation Statement

T2000 CORBA 接口可以实现以下标准的接口功能：

- MTNM V2.1 的大部分接口
- MTNM V3.0 的部分接口
- MTNM V3.5 的部分接口

1.3 特性

T2000 CORBA 接口在技术实现上具有以下特性：

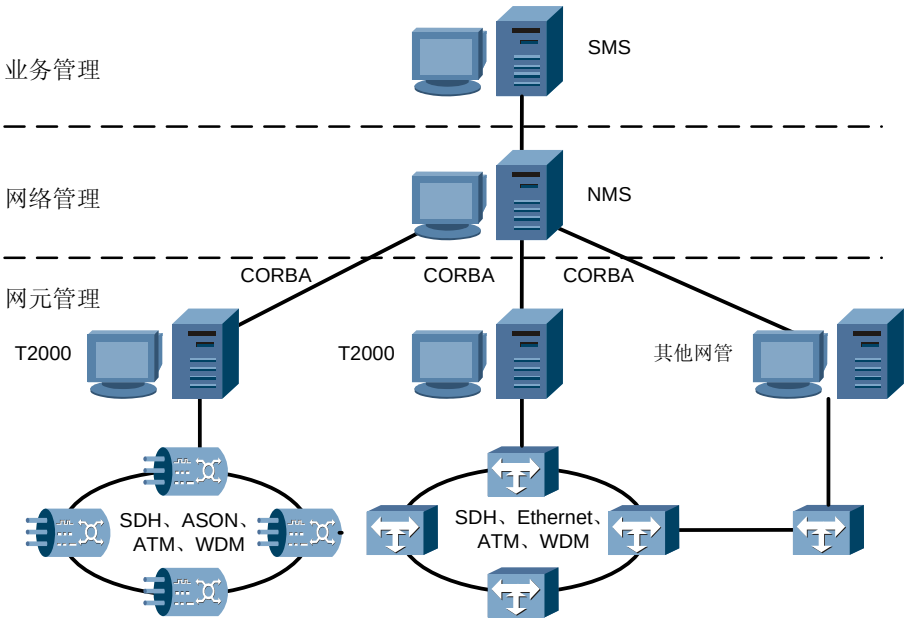
- 完全遵从 OMG（Object Management Group） CORBA 2.6 规格，支持 IIOP（Internet Inter-ORB Protocol）1.1 以及 IIOP1.2 协议。
- 采用了标准的 CORBA Naming Service1.1、Notification Service1.0。
- 目前提供的版本通过 TAO（The ACE ORB）1.4.7 实现，具有很高的效率。由于 TAO 是一个使用较广的免费 ORB（Object Request Broker）产品，因而 T2000 CORBA 接口拥有很大的成本优势，用户可以获得较高的性能价格比。
- 支持不同 ORB 之间的互通，目前已经成功实现互通的 ORB 平台产品包括：IONA Orbix2000、IONA Orbix 6.1、InterBus、JacORB、Borland VisiBroker、Borland BES 等。
- 跨操作系统平台，目前支持的平台操作系统为 Windows 和 Solaris。

- 接口支持标准 SSL 协议，通过简单的配置即可支持不同的安全访问控制模式，目前支持 IIOP 访问、SSLIOP 访问两种模式。

1.4 在综合网管中的地位

T2000 CORBA 接口在综合网管中的地位如图 1-1 所示。

图1-1 T2000 CORBA 接口在综合网管中的地位



ATM: Asynchronous Transfer Mode
SDH: Synchronous Digital Hierarchy
SMS: Service Management System
WDM: Wavelength Division Multiplexing
ASON: Automatically Switched Optical Network

T2000 CORBA 接口是网元管理层和网络管理层之间的通信枢纽。通过采用 CORBA 这一广受业界推崇的分布式对象技术，使得 iManager T2000 能够很好地适应网管综合化、跨域化的发展趋势。

2 系统特点

关于本章

本章描述内容如下表所示。

标题	内容
2.1 新增功能	T2000 CORBA 接口新增的功能介绍。
2.2 功能特色	T2000 CORBA 接口功能特色。
2.3 技术指标	T2000 CORBA 接口技术指标。

2.1 新增功能

T2000V200R005C01 和 T2000V200R004C01 版本相比新增以下功能特性：

OSS 对性能采集任务的监控

T2000V200R005C01 版本实现了对性能采集任务的管理功能。目前可以支持上层系统通过北向接口创建性能采集任务、查询性能采集任务、删除性能采集任务。

2.2 功能特色

- 严格遵循 TMF 814 IDL 接口规范。遵循该协议的多厂家设备之间可以交换管理信息，真正实现系统开放式互连。
- T2000 北向 CORBA 接口和 T2000 Server 紧密集成，可以实时反映网络信息的变化，快速响应上层网管下发的 IDL 接口命令。
- 强大的告警支持能力。能处理 T2000 接收到的所有告警，告警定位准确，能够定位到发生告警的单板、TP（Termination Point）。能够承受告警风暴。

说明

告警风暴是指单位时间（秒）内收到上百条甚至上千条的告警。

- 准确的性能监视功能。能够准确的监视不同设备对象和业务级别的性能数据，即时报告设备的运行状态。
- 方便的业务管理功能，同时支持 SDH、智能、以太网、波分业务的管理。
- 可以同时接入多个上层网管，网络信息发生变化时同时上报给所有上层网管。在与多个上层网管连接情况时，如果 CORBA 接口接收到某个上层网管的操作后，其结果会上报给所有上层网管，保证所有上层网管 信息的实时同步。
- 全面支持华为公司的 SDH、OSN、DWDM 传输设备，支持上层网管系统完成对网上各种 SDH、OSN 和 DWDM 设备的操作维护管理。

2.3 技术指标

- 通知上报技术指标：CORBA 接口每秒钟可以处理 100 个通知上报，在通知上报较多的情况下（例如告警风暴），CORBA 接口将使用缓存技术，确保不遗漏通知上报，保证上下层网管的数据一致性。缓存空间的设置可根据实际情况调节。
- 对于大多数不会下发到网元的设置操作，CORBA 接口可以在 2 秒内完成信息的设置；对于大多数下发到网元的设置操作，CORBA 接口可以在 5 秒内完成信息的设置。
- 对于小数据量查询，例如返回 MO（Managed Object）小于 1000 个，如果是单纯向网管查询，CORBA 接口在 5 秒内返回所有信息；如果需要下发网元查询，CORBA 接口返回所有信息的时间会比从网管查询的时间稍长一些，具体时间和当时的网络和网元状况有关。
- 大数据量告警查询，CORBA 接口处理速度不低于 100 条 / 秒。

3 系统功能

关于本章

本章描述内容如下表所示。

标题	内容
3.1 功能概述	T2000 CORBA 接口功能概述。
3.2 EMS 管理	T2000 CORBA 接口 EMS 管理功能介绍。
3.3 网元管理	T2000 CORBA 接口网元管理功能介绍。
3.4 设备管理	T2000 CORBA 接口设备管理功能介绍。
3.5 故障管理	T2000 CORBA 接口故障管理功能介绍。
3.6 性能管理	T2000 CORBA 接口性能管理功能介绍。
3.7 保护管理	T2000 CORBA 接口保护管理功能介绍。
3.8 维护命令	T2000 CORBA 接口维护管理功能介绍。
3.9 拓扑管理	T2000 CORBA 接口拓扑管理功能介绍。
3.10 子网连接管理	T2000 CORBA 接口子网连接管理功能介绍。
3.11 MSTP 管理	T2000 CORBA 接口 MSTP 管理功能介绍。
3.12 控制平面管理	T2000 CORBA 接口控制平面管理功能介绍。
3.13 界面直通	T2000 CORBA 接口界面直通功能介绍。
3.14 通知上报	T2000 CORBA 接口通知上报功能介绍。
3.15 安全管理	T2000 CORBA 接口安全管理功能介绍。
3.16 异常管理	T2000 CORBA 接口异常管理功能介绍。

3.1 功能概述

T2000 CORBA 接口拥有完备的功能、合理可靠的实现及很高的效率，可以平滑地北向接入遵循 TMF MTNM 系列建议的 NMS 系统，是运营商管理多技术、多供应商的异构传输网络的良好解决方案。

3.2 EMS 管理

T2000 CORBA 接口提供如下 EMS 管理功能：

- 允许 NMS 查询 T2000 管理域内所有子网的名称及其特征信息。

编程指引：

void **getAllTopLevelSubnetworks** (in unsigned long how_many, out **multiLayerSubnetwork::SubnetworkList_T** sList, out **multiLayerSubnetwork::SubnetworkIterator_I** sIt) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询 T2000 管理域内所有子网的名称。

编程指引：

void **getAllTopLevelSubnetworkNames** (in unsigned long how_many, out **globaldefs::NamingAttributesList_T** nameList, out **globaldefs::NamingAttributesIterator_I** nameIt) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询所有 T2000 管理的子网间的顶级拓扑链路名称及其特征信息。

编程指引：

void **getAllTopLevelTopologicalLinks** (in unsigned long how_many, out **topologicalLink::TopologicalLinkList_T** topoList, out **topologicalLink::TopologicalLinkIterator_I** topoIt) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询所有 T2000 管理的子网间的顶级拓扑链路名称。

编程指引：

void **getAllTopLevelTopologicalLinkNames** (in unsigned long how_many, out **globaldefs::NamingAttributesList_T** nameList, out **globaldefs::NamingAttributesIterator_I** nameIt) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定的 T2000 管理的顶级拓朴链路名称及特征信息。

编程指引：

void **getTopLevelTopologicalLink** (in **globaldefs::NamingAttributes_T** topoLinkName, out **topologicalLink::TopologicalLink_T** topoLink,) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询 T2000 的所有特征信息。包括 T2000 的名称、标识、版本等。

编程指引：

void **getEMS** (out **EMS_T** emsInfo) raises (globaldefs::ProcessingFailureException)

3.3 网元管理

T2000 CORBA 接口提供如下网元管理功能：

- 允许 NMS 查询 T2000 管理域内所有网元的名称。

编程指引：

void **getAllManagedElementNames** (in unsigned long how_many, out **globaldefs::NamingAttributesList_T** nameList, out **globaldefs::NamingAttributesIterator_I** nameIt) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询 T2000 管理域内所有网元的名称及其特征信息。

编程指引：

void **getAllManagedElements** (in unsigned long how_many, out **managedElement::ManagedElementList_T** meList, out **managedElement::ManagedElementIterator_I** meIt) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据网元名称从 T2000 查询该网元所属的子网名称及其特征信息。

编程指引：

void **getContainingSubnetworkNames** (in **globaldefs::NamingAttributes_T** managedElementName, out **globaldefs::NamingAttributesList_T** subnetNames) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据网元名称、层速率以及连接速率，从 T2000 查询该网元内满足指定层速率、指定连接速率的所有 PTP 名称及其特征信息。

编程指引：

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)

- 允许 NMS 根据网元名称从 T2000 查询该网元的特征信息。

编程指引：

void **getManagedElement** (in **globaldefs::NamingAttributes_T** managedElementName, out **managedElement::ManagedElement_T** me) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据网元名称、层速率以及连接速率，从 T2000 查询指定网元内满足指定层速率、指定连接速率的所有 PTP 名称。

编程指引：

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)

- 允许 NMS 根据 TP 名称，从 T2000 查询该 TP 的特征信息。

编程指引：

void **getTP** (in **globaldefs::NamingAttributes_T** tpName, out **terminationPoint::TerminationPoint_T** tp) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据 TP 和速率，从 T2000 查询该 TP 包含的指定速率级别的 actual TP 名称及其特征信息。"actual TP"指被各种状态的交叉连接所使用或被终结的 TP。

编程指引：

void **getContainedPotentialTPs** (in **globaldefs::NamingAttributes_T** tpName, in **transmissionParameters::LayerRateList_T** layerRateList, in unsigned long how_many, out **terminationPoint::TerminationPointList_T** tpList, out **terminationPoint::TerminationPointIterator_I** tpIt) raises (globaldefs::ProcessingFailureException)

void **getContainedInUseTPs** (in **globaldefs::NamingAttributes_T** tpName, in **transmissionParameters::LayerRateList_T** layerRateList, in unsigned long how_many, out **terminationPoint::TerminationPointList_T** tpList, out **terminationPoint::TerminationPointIterator_I** tpIt) raises (globaldefs::ProcessingFailureException)

void **getContainedCurrentTPs** (in **globaldefs::NamingAttributes_T** tpName, in **transmissionParameters::LayerRateList_T** layerRateList, in unsigned long how_many, out **terminationPoint::TerminationPointList_T** tpList, out **terminationPoint::TerminationPointIterator_I** tpIt) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据 TP 和速率，从 T2000 查询此 TP 包含的指定速率级别的 actual TP 名称。

编程指引：

void **getContainedPotentialTPNames** (in **globaldefs::NamingAttributes_T** tpName, in **transmissionParameters::LayerRateList_T** layerRateList, in unsigned long how_many, out **globaldefs::NamingAttributesList_T** nameList, out **globaldefs::NamingAttributesIterator_I** nameIt) raises (globaldefs::ProcessingFailureException)

void **getContainedInUseTPNames** (in **globaldefs::NamingAttributes_T** tpName, in **transmissionParameters::LayerRateList_T** layerRateList, in unsigned long how_many, out **globaldefs::NamingAttributesList_T** nameList, out **globaldefs::NamingAttributesIterator_I** nameIt) raises (globaldefs::ProcessingFailureException)

void **getContainedCurrentTPNames** (in **globaldefs::NamingAttributes_T** tpName, in **transmissionParameters::LayerRateList_T** layerRateList, in unsigned long how_many, out **globaldefs::NamingAttributesList_T** nameList, out **globaldefs::NamingAttributesIterator_I** nameIt) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据 TP 名称，从 T2000 查询包含该 TP 名称的 TP 特征信息列表。

编程指引：

void **getContainingTPs** (in **globaldefs::NamingAttributes_T** tpName, out **terminationPoint::TerminationPointList_T** tpList) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据 TP 名称，从 T2000 查询包含该 TP 的 TP 名称列表。

编程指引：

void **getContainingTPNames** (in **globaldefs::NamingAttributes_T** tpName, out **globaldefs::NamingAttributesList_T** tpNameList) raises (globaldefs::ProcessingFailureException)

- 通过 T2000 CORBA 接口可以设置告警屏蔽、开销字节、波分端口可调波长、TCM 检视、激光器状态、自协商模式。

编程指引：

void **setTPData** (in **subnetworkConnection::TPData_T** tpInfo, out **terminationPoint::TerminationPoint_T** modifiedTP) raises (globaldefs::ProcessingFailureException)

- 获取网元内包含的所有交叉连接的特征信息，新增包括波分交叉连接的查询（含固定交叉连接和动态交叉连接）。

编程指引：

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)

- 在指定网元上创建交叉连接，目前支持 SDH 交叉连接的创建。

编程指引：

void **createCrossConnections** (
in **subnetworkConnection::CrossConnectList_T** ccList,
out **subnetworkConnection::CrossConnectList_T** successedCCList,
out **subnetworkConnection::CrossConnectList_T** failedCCList)
raises(globaldefs::ProcessingFailureException)

- 支持批量激活指定的 SDH 交叉连接。（建议：每次查询的数量 how_many 不要超过 200 条。）

编程指引：

void **activateCrossConnections** (
in **subnetworkConnection::CrossConnectList_T** ccList,
out **subnetworkConnection::CrossConnectList_T** successedCCList,
out **subnetworkConnection::CrossConnectList_T** failedCCList)
raises(globaldefs::ProcessingFailureException)

- 支持批量去激活指定的 SDH 交叉连接。（建议：每次查询的数量 how_many 不要超过 200 条。）

编程指引：

void deactivateCrossConnections (
in subnetworkConnection::CrossConnectList_T ccList,
out **subnetworkConnection::CrossConnectList_T** successedCCList,
out **subnetworkConnection::CrossConnectList_T** failedCCList)
raises(globaldefs::ProcessingFailureException)

- 支持批量删除指定的 SDH 交叉连接。

编程指引：

void **deleteCrossConnections** (
in **subnetworkConnection::CrossConnectList_T** ccList,
out **subnetworkConnection::CrossConnectList_T** successedCCList,
out **subnetworkConnection::CrossConnectList_T** failedCCList)
raises(globaldefs::ProcessingFailureException)

- 查询指定网元的静态信息。

编程指引：

void **getNEStaticInfo**(
in **globaldefs::NamingAttributes_T** managedElementName,
in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** staticInfoList,
out **globaldefs::NamingAttributesIterator_I** staticInfoIt)
raises(globaldefs::ProcessingFailureException)

3.4 设备管理

T2000 CORBA 接口提供如下设备管理功能：

- 允许 NMS 查询指定网元或设备容器中包含的所有子设备的名称及其特征信息。

编程指引：

void **getAllEquipment** (in **globaldefs::NamingAttributes_T** meOrHolderName, in unsigned long how_many, out **EquipmentOrHolderList_T** eqList, out **EquipmentOrHolderIterator_I** eqIt) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定网元或设备容器中包含的所有子设备的名称。

编程指引：

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)

- 允许 NMS 查询包含指定 PTP 的设备的名称及其特征信息。

编程指引：

void **getAllSupportingEquipment** (in **globaldefs::NamingAttributes_T** ptpName, out **EquipmentOrHolderList_T** eqList) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询包含指定 PTP 的设备的名称。

编程指引：

void **getAllSupportingEquipmentNames** (in **globaldefs::NamingAttributes_T** ptpName, out **globaldefs::NamingAttributesList_T** nameList) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定设备容器直接包含的所有子设备的名称及其特征信息。

编程指引：

void **getContainedEquipment** (in **globaldefs::NamingAttributes_T** equipmentHolderName, out **EquipmentOrHolderList_T** equipmentOrHolderList) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定设备或设备容器的特征信息

编程指引：

void **getEquipment** (in **globaldefs::NamingAttributes_T** equipmentOrHolderName, out **EquipmentOrHolder_T** equip) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询机房、机柜以及子架的物理位置信息

编程指引：

void **getPhysicalLocationInfo**(out **PhysicalLocationInfoList_T** phyLocationInfoList) raises(globaldefs::ProcessingFailureException)

- 允许 NMS 查询 T2000 支持的所有单板的静态信息

编程指引：

void **getEquipmentStaticInfo** (in EquipmentObjectTypeList_T typeList, in unsigned long how_many, out **globaldefs::NamingAttributesList_T** staticInfoList, out **globaldefs::NamingAttributesIterator_I** staticInfoIt) raises (globaldefs::ProcessingFailureException)

- 获取设备支持的终结点

允许 NMS 查询指定单板所支持的 TPs（PTPs）。这将使得网络管理应用能解释硬件中断和确定业务影响（保护丢失和整个功能丢失）。可用于故障定位和网络规划。

编程指引：

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)

- 获取设备支持的终结点名称

允许 NMS 查询指定单板所支持的 TPs（PTPs）。这将使得网络管理应用能解释硬件中断和确定业务影响（保护丢失和整个功能丢失）。可用于故障定位和网络规划。

编程指引：

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)

- 允许 NMS 在网元中删除单板。

编程指引：

void **unprovisionEquipment** (in **globaldefs::NamingAttributes_T** equipmentName) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 在网元中创建单板。

编程指引：

void **provisionEquipment** (in **EQTCreateData_T** equipmentCreateData, out **Equipment_T** createdEquipment) raises (globaldefs::ProcessingFailureException)

3.5 故障管理

T2000 CORBA 接口提供如下故障管理功能：

- NMS 可以通过 T2000 CORBA 接口向 T2000 查询该 T2000 控制下的满足约束条件的告警和 TCA（Threshold Crossing Alarm）。

编程指引：

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)

- NMS 可以通过 T2000 CORBA 接口向 T2000 查询指定网元的满足约束条件的告警和 TCA。（建议：每次输入查询告警的数量 how_many 不要超过 500 条。）

编程指引：

void **getAllActiveAlarms** (in **globaldefs::NamingAttributes_T** meName, 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)

- NMS 通过 CORBA 接口查询 T2000 的系统告警。（建议：每次输入查询告警的数量 how_many 不要超过 500 条。）

编程指引：

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)

- 通过 CORBA Notification Service，NMS 可以向 T2000 登记，以接收告警事件或 TCA。（建议：每次输入查询告警的数量 how_many 不要超过 500 条。）

- 通过 T2000 CORBA 接口可以设置单板/子架的告警上报开关。

编程指引：

void **setAlarmReportingOn** (in **globaldefs::NamingAttributes_T** equipmentOrHolderName) raises (globaldefs::ProcessingFailureException)

void **setAlarmReportingOff** (in **globaldefs::NamingAttributes_T** equipmentOrHolderName) raises (globaldefs::ProcessingFailureException)

- 通过 T2000 CORBA 接口可以设置 TP 的告警上报开关。

编程指引：

void **setTPData** (in **subnetworkConnection::TPData_T** tpInfo, out **terminationPoint::TerminationPoint_T** modifiedTP) raises (globaldefs::ProcessingFailureException)

- 通过 CORBA Notification Service，T2000 可以主动向 NMS 发送告警事件和 TCA。
- 通过 T2000 CORBA 接口 NMS 可以向 T2000 核对当前告警。（建议：每次核对告警的数量不要超过 500 条。）

编程指引：

void **checkActiveAlarms** (in **notifications::EventList_T** activeEventList, out **notifications::EventList_T** clearedEventList) raises (globaldefs::ProcessingFailureException)

- NMS 通过 T2000 CORBA 接口可以查询指定对象（如：TP，Equipment）上的告警。（建议：每次输入查询告警的数量 how_many 不要超过 500 条。）

编程指引：

void **getActiveAlarms** (in **globaldefs::NamingAttributesList_T** objectNameList, 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)

3.6 性能管理

T2000 CORBA 接口提供如下设备管理功能：

- PM 数据收集的使能/禁止
- 完全支持 NMS 使能/禁止网元的 PM（Performance Monitor）数据收集。
- 支持使能/禁止 TP 的 15 分钟和 24 小时 PM 数据收集。

编程指引：

void **disablePMData** (in **PMTPSelectList_T** pmTPSelectList, out **PMTPSelectList_T** failedTPSelectList) raises (globaldefs::ProcessingFailureException)

void **enablePMData** (in **PMTPSelectList_T** pmTPSelectList, out **PMTPSelectList_T** failedTPSelectList) raises (globaldefs::ProcessingFailureException)

- 支持 PM 寄存器清空/复位。

编程指引：

void **clearPMData** (in **PMTPSelectList_T** pmTPSelectList, out **PMTPSelectList_T** failedTPSelectList) raises (globaldefs::ProcessingFailureException)

- NMS 可以通过 T2000 CORBA 接口向 T2000 查询网元在指定层速率支持的 PM 参数集。

编程指引：

void **getMEPMcapabilities** (in **globaldefs::NamingAttributes_T** meName, in **transmissionParameters::LayerRate_T** layerRate, out **PMParameterList_T** pmParameterList) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 向 T2000 查询在 T2000 中保留多少小时的性能数据；如果 T2000 不保留性能数据，则返回网元保留性能数据的时间。

编程指引：

void **getHoldingTime** (out **HoldingTime_T** holdingTime) raises (globaldefs::ProcessingFailureException)

- 门限设置

允许 NMS 设置 TP 的 TCA 参数。该操作可以修改 TP-layerRate 测量点上的 15 分钟/24 小时 TCA 门限值。

编程指引：

void **setTCATPParameter** (in **globaldefs::NamingAttributes_T** tpName, inout **TCAParameters_T** tcaParameters) raises (globaldefs::ProcessingFailureException)

- 允许 T2000 自动发送越限告警通知。
- 门限查询

NMS 可以查询指定层速率的 TP 测量点的 TCA 门限当前值。

编程指引：

void **getTCATPParameter** (in **globaldefs::NamingAttributes_T** tpName, in **transmissionParameters::LayerRate_T** layerRate, in **Granularity_T** granularity, out **TCAParameters_T** tcaParameter) raises (globaldefs::ProcessingFailureException)

- 历史 PM 数据查询

查询历史性能数据。通过 T2000 CORBA 接口，NMS 可以请求 T2000 将指定测量点的历史性能数据存入一个文件，并通过 FTP（File Transfer Protocol）方式把此文件发送到请求中指定的目的地。

编程指引：

void **getHistoryPMData** (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)

- 当前 PM 数据查询

支持 NMS 查询一系列 TP 测量点的当前性能数据。

编程指引：

void **getAllCurrentPMData** (in **PMTPSelectList_T** pmTPSelectList, in **PMParameterNameList_T** pmParameters, in unsigned long how_many, out **PMDataList_T** pmDataList, out **PMDataIterator_I** pmIt) raises (globaldefs::ProcessingFailureException)

- 性能事件上报状态查询

允许 NMS 查询一系列 TP 检测点的性能事件状态(是否自动上报到 EMS 网管)。

编程指引：

void **getPMState** (in **PMTPSelectList_T** pmTPSelectList, in **PMParameterNameList_T** pmParameters, in unsigned long how_many, out **PMStateList_T** pmStateList, out **PMStateIterator_I** pmSateIt) raises (globaldefs::ProcessingFailureException)

- 性能定时采集任务创建

支持 NMS 创建定时采集任务。该任务用来描述周期性采集性能数据的一些信息。

编程指引:

void **createPMCollectionTask** (in string taskName,in string destination, in string userName, in string password, in string emsUserName, in globaldefs::NamingAttributesList_T pmTPSelectList, in string period, in string startTime, in string endTime, in boolean forceUpload) raises(globaldefs::ProcessingFailureException)

- 性能定时采集任务删除

支持 NMS 按性能采集任务名称删除一个采集任务的对象。任务对象被删除后，相应的性能采集任务将结束。

编程指引:

void **deletePMCollectionTask** (in **CollectTaskNameList_T** taskNameList, in string emsUserName) raises(globaldefs::ProcessingFailureException)

- 性能定时采集任务查询

支持 NMS 按性能采集任务名称查询性能采集任务，输入为空则默认返回全部任务。

编程指引:

void **selectPMCollectionTask** (in **CollectTaskNameList_T** taskNameList, in string emsUserName, out **CollectTaskInfoList_T** taskInfoList) raises(globaldefs::ProcessingFailureException)

3.7 保护管理

T2000 CORBA 接口提供了全面的保护管理功能。NMS 借此能够查询复用段保护组、设备保护组以及波分保护组信息，执行保护倒换。在设备发生倒换后，T2000 向 NMS 上报保护倒换通知。包括：

- 查询指定网元中的所有复用段保护组的名字及其特征信息。

编程指引：

void **getAllProtectionGroups** (in **globaldefs::NamingAttributes_T** meName, in unsigned long how_many, out **ProtectionGroupList_T** pgList, out **ProtectionGroupIterator_I** pgpIt) raises (globaldefs::ProcessingFailureException)

- 查询 T2000 中指定复用段保护组的特征信息。

编程指引：

void **getProtectionGroup** (in **globaldefs::NamingAttributes_T** pgName, out **protection::ProtectionGroup_T** protectionGroup) raises (globaldefs::ProcessingFailureException)

- 查询复用段保护组中所有携带"不可预占无保护额外业务"的 CTP 的名字。

编程指引：

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)

- 查询指定 TP 的相邻 TP。

编程指引：

void **getAdjacentTPs** (in **globaldefs::NamingAttributes_T** tpName, out **globaldefs::NamingAttributesList_T** tpNameList) raises (globaldefs::ProcessingFailureException)

- 查询复用段保护组中所有携带"可预占无保护额外业务"的 CTP 的名字。

编程指引：

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)

- 查询复用段保护组中所有携带"被保护业务"的 CTP 的名字。

编程指引：

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)

- 查询 SNCP 或者复用段保护组的倒换状态。

编程指引：

void **retrieveSwitchData** (in **globaldefs::NamingAttributes_T** reliableSinkCtpOrGroupName, out **protection::SwitchDataList_T** switchData) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 对指定的复用段保护组或 SNCP 保护组 TP 执行外部倒换命令。

编程指引：

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)

- 允许 NMS 根据网元名称查询网元内所有的设备保护组的特征信息。

编程指引：

void **getAllEProtectionGroups**(in **globaldefs::NamingAttributes_T** meName,in unsigned long how_many,out **EProtectionGroupList_T** epgpList,out **EProtectionGroupIterator_I** epgpIt)

raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据设备保护组的名称，查询该保护组的特征信息。

编程指引：

void **getEProtectionGroup**(in **globaldefs::NamingAttributes_T** ePGPname,out **protection::EProtectionGroup_T** eProtectionGroup)

raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据波分网元名称查询该波分网元内所有设备波分保护组的特征信息。

编程指引：

void **getAllWDMProtectionGroups**(in **globaldefs::NamingAttributes_T** meName,in unsigned long how_many,out **WDMProtectionGroupList_T** wpgpList,out **WDMProtectionGroupIterator_I** wpgpIt)

raises(globaldefs::ProcessingFailureException)

- 允许 NMS 根据波分保护组的名称查询该波分保护组的特征信息。

编程指引：

void **getWDMProtectionGroup**(in **globaldefs::NamingAttributes_T** wpgpName,out **protection::WDMProtectionGroup_T** wProtectionGroup)

raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据设备保护组的名称，查询该设备保护组的倒换信息。

编程指引：

void **retrieveESwitchData**(in **globaldefs::NamingAttributes_T** ePGPName,out **protection::ESwitchDataList_T** eSwitchDataList)

raises(globaldefs::ProcessingFailureException)

- 允许 NMS 根据波分保护组的名称，查询该波分保护组的倒换信息。

编程指引：

void **retrieveWDMSwitchData**(in **globaldefs::NamingAttributes_T** wpgpName,out **protection::WDMSwitchDataList_T** wSwitchDataList)

raises(globaldefs::ProcessingFailureException)

- 允许 NMS 对波分保护组执行外部倒换命令。

编程指引：

void **performWDMProtectionCommand**(in **ProtectionCommand_T** protectionCommand, in **globaldefs::NamingAttributes_T** wpgpName,

in **globaldefs::NamingAttributes_T** fromTp, in **globaldefs::NamingAttributes_T** toTp, out **protection::WDMSwitchData_T** wSwitchData) raises(globaldefs::ProcessingFailureException)

- 保护倒换通知上报。

保护组属性或状态变更的通知上报。

3.8 维护命令

T2000 CORBA 接口在 TP 上支持以下操作维护功能：

- 允许 NMS 设置和撤销对 TP 的维护操作。

编程指引：

void **performMaintenanceOperation** (in **CurrentMaintenanceOperation_T** maintenanceOperation, in **MaintenanceOperationMode_T** maintenanceOperationMode) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询 T2000 侧是否已发起了对 TP 的维护操作，该查询可以被 PTP、CTP（VC4）和 ME 对象支持。

编程指引：

void **getActiveMaintenanceOperations** (in **globaldefs::NamingAttributes_T** tpOrMeName, in unsigned long how_many, out **CurrentMaintenanceOperationList_T** currentMaintenanceOpeationList, out **CurrentMaintenanceOperationIterator_I** cmoIt) raises (globaldefs::ProcessingFailureException)

具体支持的维护操作如下：

- 设备环回（外环回）
- 终端环回（内环回）
- AIS 告警插入
- RDI 告警插入

- 启动 PRBS 测试。

编程指引：

void **enablePRBSTest**(in **PRBSTestParameterList_T** testParaList,

out **globaldefs::NamingAttributesList_T** failedTPList)

raises (globaldefs::ProcessingFailureException)

- 停止 PRBS 测试。

编程指引：

void **disablePRBSTest**(in **globaldefs::NamingAttributesList_T** tpNameList,

out **globaldefs::NamingAttributesList_T** failedTPList)

raises (globaldefs::ProcessingFailureException)

- 查询 PRBS 测试结果。

编程指引：

void **getPRBSTestResult**(in **globaldefs::NamingAttributesList_T** tpNameList,

out **PRBSTestResultList_T** resultList)

raises (globaldefs::ProcessingFailureException)

3.9 拓扑管理

T2000 CORBA 接口拓扑管理具有以下功能：

- 允许 NMS 查询网管拓扑视图中的网元和子网信息，主要是坐标信息。

编程指引：

void **getTopoSubnetworkViewInfo**(in unsigned long how_many,
out **NodeList_T** nodeList,out **NodeIterator_I** NodeIt)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 查询网管保护子网视图中的网元信息，主要是坐标信息。

编程指引：

void **getProtectSubnetworkViewInfo**(in unsigned long how_many,
out **NodeList_T** nodeList,out **NodeIterator_I** NodeIt)
raises(globaldefs::ProcessingFailureException)

3.10 子网连接管理

T2000 CORBA 接口子网连接管理具有以下功能：

- 允许 NMS 根据子网名称及子网连接速率从 T2000 查询该子网内满足指定速率的所有子网连接名称及其特征信息。

编程指引：

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)

- 允许 NMS 根据子网名称及子网连接速率从 T2000 查询该子网内满足指定速率的所有子网连接名称。

编程指引：

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)

- 允许 NMS 根据指定的 TP（Termination Point）及子网连接速率从 T2000 查询所有相关的子网连接名称及其特征信息：（a）指定一个 PTP（Physical Termination Point），返回所有的 SNC（Subnetwork Connection）。这些 SNC 必须满足指定的连接速率，并且经过该 PTP 包含的 CTP（Connection Termination Point）。（b）指定一个 CTP，返回所有的 SNC。这些 SNC 必须满足指定的连接速率，并且经过该 CTP 或者经过该 CTP 包含的 CTP。

编程指引：

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)

- 允许 NMS 根据指定的 TP 及层速率，从 T2000 查询所有相关的子网连接名称。（建议：每次查询的数量 how_many 不要超过 200 条。）

编程指引：

void **getAllSubnetworkConnectionNamesWithTP** (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)

- 允许 NMS 根据指定的 UserLabel，从 T2000 查询相关的子网连接特征信息。

编程指引：

void **getSNCsByUserLabel** (in string userLabel, out **subnetworkConnection::SubnetworkConnectionList_T** sncList) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据子网连接名称，从 T2000 查询该子网连接的特征信息。（建议：每次查询的数量 how_many 不要超过 200 条。）

编程指引：

void **getSNC** (in **globaldefs::NamingAttributes_T** sncName, out **subnetworkConnection::SubnetworkConnection_T** snc) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据子网连接名称，从 T2000 查询该子网连接的路由信息。

编程指引：

void **getRoute** (in **globaldefs::NamingAttributes_T** sncName, in boolean includeHigherOrderCCs, out **subnetworkConnection::Route_T** route) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据子网连接名称，从 T2000 查询该子网连接的路由以及该 SNC 经过的拓扑连接信息。

编程指引：

void **getRouteAndTopologicalLinks** (in **globaldefs::NamingAttributes_T** sncName, out **subnetworkConnection::Route_T** route, out **topologicalLink::TopologicalLinkList_T** topologicalLinkList) raises (**globaldefs::ProcessingFailureException**)

- 允许 NMS 根据指定的子网连接名称列表，从 T2000 查询这些子网连接的路由名称及其特征信息。

编程指引：

void **getRoutes** (in **globaldefs::NamingAttributesList_T** sncNameList, in boolean includeHigherOrderCCs, out **subnetworkConnection::RouteInfoList_T** routeInfoList) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据指定的 nativeEMSName，从 T2000 查询相关的子网连接的特征信息，支持通配符方式的查询。

编程指引：

void **getSNCsByNativeEmsName** (in string nativeEmsName, in unsigned long how_many, out **subnetworkConnection::SubnetworkConnectionList_T** sncList, out **subnetworkConnection::SNCIterator_I** sncIt) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 指定子网连接经过的终端点查询相关的子网连接的特征信息。

编程指引：

void **getSNCsByEndObjectName** (in **globaldefs::NamingAttributes_T** aEndObjectName, in **globaldefs::NamingAttributes_T** zEndObjectName, in **transmissionParameters::LayerRateList_T** connectionRateList, out **subnetworkConnection::SubnetworkConnectionList_T** sncList) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 一次查询多条指定的子网连接的特征信息。（建议：每次查询的数量 how_many 不要超过 200 条。）

编程指引：

void **getSNCs** (in **globaldefs::NamingAttributesList_T** sncNameList, out **subnetworkConnection::SubnetworkConnectionList_T** sncList) raises (globaldefs::ProcessingFailureException)z

- 允许 NMS 根据子网名称从 T2000 查询该子网的特征信息。（建议：每次查询的数量 how_many 不要超过 200 条。）

编程指引：

void **getMultiLayerSubnetwork** (in **globaldefs::NamingAttributes_T** subnetName, out **MultiLayerSubnetwork_T** subnetwork) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 在 T2000 中创建一个规划的子网连接。

如果请求成功，则 T2000 会创建一个子网连接对象，但是不会在网元设备上创建和这个子网连接相关的任何交叉连接。如果创建成功，但未激活，SNC 状态为 pending。

NMS 通过 CORBA 接口请求 T2000 创建一个子网连接时，需要提供下列信息：

- 用户标签
- 用户标签是否唯一
- 所有者
- 方向
- 静态保护级别
- 保护努力指示
- 是否重算路由指示
- 网络层计算路由指示
- SNC 类型
- 层速率
- 路由约束
- 路由约束是否完整指示
- A 端 CTP(s)
- Z 端 CTP(s)
- 附加信息
- 最大可容忍业务中断级别
- EMS 自由度指示

编程指引：

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)

- NMS 可以请求 T2000 激活一个指定的 SNC。如果激活成功，就表明 T2000 已经下发命令，使所有组成子网连接的交叉连接已经在网元设备上就位。此时 SNC 状态迁移至 active。

编程指引：

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)

- NMS 给出 SNC 数据，请求 T2000 依据此数据创建并激活一个 SNC。如果请求成功，就表明 T2000 创建了一个表示 SNC 的对象，并已经下发命令，使所有组成子网连接的交叉连接已经在网元上就位。成功创建并激活的 SNC 状态迁移至 active。

编程指引：

void **createAndActivateSNC** (in **subnetworkConnection::SNCCreateData_T** createData, 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)

- NMS 给出子网连接的名字，请求 T2000 去激活该 SNC。如果请求成功，就表明 T2000 已经下发命令，在网元上删除所有构成此 SNC 的交叉连接，但是 T2000 却保留表示该 SNC 的对象。成功去激活的 SNC 状态迁移至 pending。

编程指引：

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)

- NMS 给出子网连接的名字，删除子网连接。如果请求成功，T2000 应删除表示子网连接的对象。如果任何一个构成子网连接的交叉连接还在网元中存在，T2000 将拒绝该请求。也就是说，在 T2000 允许删除子网连接之前，必须完全成功地去激活该子网连接。

编程指引：

void **deleteSNC** (in **globaldefs::NamingAttributes_T** sncName, in **EMSFreedomLevel_T** emsFreedomLevel) raises (globaldefs::ProcessingFailureException)

- NMS 给出 SNC 数据，请求 T2000 依据此数据去激活并删除一个 SNC。如果请求成功，就表明 T2000 已经下发命令，在网元上删除所有构成此 SNC 的交叉连接，并在 T2000 中删除表示该子网连接的对象。

编程指引：

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)

- NMS 给出要创建的 SNC 数据，T2000 经过对当前资源的计算后，判断能否创建该 SNC。该操作不会在 T2000 和网元上创建任何的对象。

编程指引：

void **checkValidSNC** (in **subnetworkConnection::SNCCreateData_T** createData, in **subnetworkConnection::TPDataList_T** tpsToModify, in boolean considerResources, out boolean valid) raises (globaldefs::ProcessingFailureException)

- NMS 给出需要修改的子网连接名称和的数据信息，请求 T2000 进行修改。如果请求成功，表明 T2000 已修改成功，将返回修改后的子网连接信息。

编程指引：

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,
out **subnetworkConnection::SubnetworkConnection_T** newSNC,
out string errorReason) raises (globaldefs::ProcessingFailureException)

- NMS 给出要设置相关联(去关联)的两条子网连接的名称，请求 T2000 将这两条子网连接修改并激活为相关联(去关联)的子网连接。该操作只对智能子网连接才有意义。

编程指引：

void **setConjunctionSNC**(in **globaldefs::NamingAttributes_T** sncName1,
in **globaldefs::NamingAttributes_T** sncName2, in boolean operate)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 根据子网名称、层速率和连接速率，从 T2000 查询该子网内满足指定层速率、连接速率的所有边界点（PTP 集合）名称及其特征信息。

编程指引：

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)

- 允许 NMS 根据子网名称、层速率和连接速率，从 T2000 查询该子网内满足指定层速率、连接速率的所有边界点（PTP 集合）名称。

编程指引：

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)

- 允许 NMS 查询子网内所有网元的名称。

编程指引：

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)

- 允许 NMS 查询子网内所有网元的名称及其特征信息。

编程指引：

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)

- 允许 NMS 根据子网名称从 T2000 查询该子网内网元间的拓扑链路名称及其特征信息。

编程指引：

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)

- 允许 NMS 根据子网名称从 T2000 查询该子网内网元间的拓扑逻辑连接名称。

编程指引：

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)

- 允许 NMS 根据拓扑逻辑连接名称从 T2000 查询该连接的详细信息。

编程指引：

void **getTopologicalLink** (in **globaldefs::NamingAttributes_T** topoLinkName, out **globaldefs::NamingAttributesList_T** nameList, out **topologicalLink::TopologicalLink_T** topoLink) raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据光网元名称从 T2000 查询该网元内所有拓扑连接的信息。

编程指引：

void **getAllInternalTopologicalLinks**(in **globaldefs::NamingAttributes_T** meName, in unsigned long how_many, out **topologicalLink::TopologicalLinkList_T** topoList,out **topologicalLink::TopologicalLinkIterator_I** topoIt)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 根据光网元名称从 T2000 查询该网元内所有拓扑连接的名称。

编程指引：

void **getAllInternalTopologicalLinkNames**(in **globaldefs::NamingAttributes_T** meName, in unsigned long how_many, out **globaldefs::NamingAttributesList_T** nameList, out **globaldefs::NamingAttributesIterator_I** nameIt)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 根据 SNC 名称进行 SNC 切换。

编程指引：

void **swapSNC**(in **globaldefs::NamingAttributes_T** nameOfSNCtoBeDeactivated,in **globaldefs::NamingAttributes_T** nameOfSNCtoBeActivated,in **EMSFreedomLevel_T** emsFreedomLevel,in **subnetworkConnection::GradesOfImpact_T** tolerableImpact,inout **subnetworkConnection::TPDataList_T** tpsToModify,out **subnetworkConnection::SNCState_T** stateOfActivatedSNC,out string errorReason)raises (globaldefs::ProcessingFailureException)

3.11 MSTP 管理

3.11.1 MSTP 存量管理

T2000 CORBA 接口提供对 MSTP 设备的存量管理功能，该部分功能参考 TMF814 建议，根据华为自定义 IDL 开发，实现对 Mac 端口、VCTrunk 端口、RPR 端口、ATM 端口、ATMTrunk 端口、虚拟网桥、VLAN 转发过滤表、VCTrunk 通道绑定等存量信息的管理功能。

- 允许 NMS 查询指定网元内所有的 MSTP 终端点（包括 Mac 端口、VCTrunk 端口、RPR 端口、逻辑端口、ATM 端口、ATMTrunk 端口）的特征信息。

编程指引：

void **getAllMstpEndPoints**(in **globaldefs::NamingAttributes_T** meName, in **HW_mstpInventory::HW_MSTPEndPointTypeList_T** typeList, in unsigned long how_many, out **HW_mstpInventory::HW_MSTPEndPointList_T** endPointList,

out **HW_mstpInventory::HW_MSTPEndPointIterator_I** endPointIt)raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定网元内所有的 MSTP 终端点（包括 Mac 端口、VCTrunk 端口、RPR 端口、逻辑端口、ATM 端口、ATMTrunk 端口）的名字。

编程指引：

void **getAllMstpEndPointNames**(in **globaldefs::NamingAttributes_T** meName,

in **HW_mstpInventory::HW_MSTPEndPointTypeList_T** typeList,

in unsigned long how_many,

out **globaldefs::NamingAttributesList_T** nameList,

out **globaldefs::NamingAttributesIterator_I** nameIt)

raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定的 MSTP 终端点的特征信息。

编程指引：

void **getMstpEndPoint**(in **globaldefs::NamingAttributes_T** endPointName,

out **HW_mstpInventory::HW_MSTPEndPoint_T** endPoint)

raises (globaldefs::ProcessingFailureException)

- 允许 NMS 设置指定的 MSTP 终端点的特征信息。

编程指引：

void **setMstpEndPoint**(in **globaldefs::NamingAttributes_T** endPointName,

in **transmissionParameters::LayeredParameterList_T** paraList,

out **HW_mstpInventory::HW_MSTPEndPoint_T** endPoint)

raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询网元内所有虚拟网桥的特征信息。

编程指引：

void **getAllVBs**(in **globaldefs::NamingAttributes_T** meName,

in unsigned long how_many,

out **HW_mstpInventory::HW_VirtualBridgeList_T** vbList,

out **HW_mstpInventory::HW_VirtualBridgeIterator_I** vbIt)

raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询网元内所有虚拟网桥的名字。

编程指引：

void **getAllVBNames**(in **globaldefs::NamingAttributes_T** meName,

in unsigned long how_many,

out **globaldefs::NamingAttributesList_T** nameList,

out **globaldefs::NamingAttributesIterator_I** nameIt)

raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定的虚拟网桥的特征信息。

编程指引：

void **getVirtualBridge**(in **globaldefs::NamingAttributes_T** vbName,

out **HW_mstpInventory::HW_VirtualBridge_T** vb)

raises (globaldefs::ProcessingFailureException)

- 允许 NMS 创建虚拟网桥。

编程指引：

void **createVirtualBridge**(in **globaldefs::NamingAttributes_T** equipmentName,
in unsigned short vbId,
in string vbName,
out **HW_mstpInventory::HW_VirtualBridge_T** vb)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 删除指定的虚拟网桥。

编程指引：

void **deleteVirtualBridge**(in **globaldefs::NamingAttributes_T** vbName)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定虚拟网桥内所有 VLAN 的特征信息。

编程指引：

void **getAllVLANs**(in **globaldefs::NamingAttributes_T** vbName,
in unsigned long how_many,
out **HW_mstpInventory::HW_VirtualLANList_T** vlanList,
out **HW_mstpInventory::HW_VirtualLANIterator_I** vlanIt)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定虚拟网桥内所有 VLAN 的名字。

编程指引：

void **getAllVLANNames**(in **globaldefs::NamingAttributes_T** vbName,
in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定 VLAN 的特征信息。

编程指引：

void **getVLAN**(in **globaldefs::NamingAttributes_T** vlanName,
out **HW_mstpInventory::HW_VirtualLAN_T** vlan)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 创建 VLAN。

编程指引：

void **createVLAN**(in **globaldefs::NamingAttributes_T** vbName,
in unsigned short vlanId,
in **globaldefs::NamingAttributesList_T** forwardTPList,
out **HW_mstpInventory::HW_VirtualLAN_T** vlan)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 删除指定的 VLAN。

编程指引：

void **deleteVLAN**(in **globaldefs::NamingAttributes_T** vlanName)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 动态增加 VLAN 的转发过滤端口。

编程指引：

void **addVLANForwardPort**(in **globaldefs::NamingAttributes_T** vlanName,
in **globaldefs::NamingAttributesList_T** forwardTPList,
out **HW_mstpInventory::HW_VirtualLAN_T** vlan)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 动态减少 VLAN 的转发过滤端口。

编程指引：

void **delVLANForwardPort**(in **globaldefs::NamingAttributes_T** vlanName,
in **globaldefs::NamingAttributesList_T** forwardTPList,
out **HW_mstpInventory::HW_VirtualLAN_T** vlan)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 设置 VLAN 的属性。

编程指引：

void **setVLANData**(in **globaldefs::NamingAttributes_T** vlanName,
in **globaldefs::NVSList_T** paraList,
out **HW_mstpInventory::HW_VirtualLAN_T** vlan)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询 VCTrunk 端口的通道绑定信息。

编程指引：

void **getBindingPath**(in **globaldefs::NamingAttributes_T** endPointName,
out **HW_mstpInventory::HW_MSTPBindingPathList_T** bindingPathList)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 动态增加 VCTrunk 端口的绑定通道。

编程指引：

void **addBindingPath**(in **globaldefs::NamingAttributes_T** endPointName,
in **terminationPoint::Directionality_T** bindingDirect,
in **globaldefs::NamingAttributesList_T** pathList,
out **HW_mstpInventory::HW_MSTPBindingPathList_T** bindingPathList)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 动态减少 VCTrunk 端口的绑定通道。

编程指引：

void **delBindingPath**(in **globaldefs::NamingAttributes_T** endPointName,
in **terminationPoint::Directionality_T** bindingDirect,
in **globaldefs::NamingAttributesList_T** pathList,
out **HW_mstpInventory::HW_MSTPBindingPathList_T** bindingPathList)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询 VCTrunk 端口的 LCAS 协议使能状态。

编程指引：

void **getLCASState**(in **globaldefs::NamingAttributes_T** endPointName,
out boolean enableState)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 设置 VCTrunk 端口的 LCAS 协议使能状态。

编程指引：

void **setLCASState**(in **globaldefs::NamingAttributes_T** endPointName,
in boolean enableState)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定的 ATM/ATMTrunk 端口包含的所有的正在使用的（表示已经被 ATM 交叉连接占用或者被映射的 CTP，如 VP CTP 包含的某一个 VC CTP 被连接占用，该 VP CTP 也被视为正在使用的 CTP）ATM VP/VC CTP 的名字。

编程指引：

void **getAllContainedInUseTPNames**(in **globaldefs::NamingAttributes_T** endPointName,
in **transmissionParameters::LayerRateList_T** layerRateList,
in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 创建 QoS 规则。

编程指引：

void **createQosRule**(in **globaldefs::NamingAttributes_T** equipmentName,
in **HW_mstpInventory::HW_QosType_T** qosType,
in **globaldefs::NVSList_T** paraList,
out **HW_mstpInventory::HW_QosRule_T** qosRule)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 设置 QoS 规则。

编程指引：

void **setQosRule**(in **globaldefs::NamingAttributes_T** qosRuleName,
in **globaldefs::NVSList_T** paraList,
out **HW_mstpInventory::HW_QosRule_T** qosRule)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 创建流。

编程指引：

void **createFlow**(in **globaldefs::NamingAttributes_T** equipmentName,
in **globaldefs::NVSList_T** paraList,
out **HW_mstpInventory::HW_Flow_T** flow)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询所有 QoS 规则的信息。

编程指引：

void **getAllQosRules**(in **globaldefs::NamingAttributes_T** meName,
in unsigned long how_many,
out **HW_mstpInventory::HW_QosRuleList_T** qosRuleList,
out **HW_mstpInventory::HW_QosRuleIterator_I** qosRuleIt)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询所有的 QoS 规则的名称。

编程指引：

void **getAllQosRuleNames**(in **globaldefs::NamingAttributes_T** meName,
in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据 QoS 规则的名称，查询该 QoS 规则的详细信息。

编程指引：

void **getQosRule**(in **globaldefs::NamingAttributes_T** qosRuleName,
out **HW_mstpInventory::HW_QosRule_T** qosRule)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据 QoS 规则的名称，删除该 QoS 规则。

编程指引：

void **deleteQosRule**(in **globaldefs::NamingAttributes_T** qosRuleName)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询所有的流的详细信息。

编程指引：

void **getAllFlows**(in **globaldefs::NamingAttributes_T** meName,
in unsigned long how_many,
out **HW_mstpInventory::HW_FlowList_T** flowList,
out **HW_mstpInventory::HW_FlowIterator_I** flowIt)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询所有流的名称。

编程指引：

void **getAllFlowNames**(in **globaldefs::NamingAttributes_T** meName,
in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 根据流的名称，查询该流的详细信息。

编程指引：

void **getFlow**(in **globaldefs::NamingAttributes_T** flowName,
out **HW_mstpInventory::HW_Flow_T** flow)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 设置流和 QoS 的绑定关系。

编程指引：

void **setFlow**(in **globaldefs::NamingAttributes_T** flowName,
in **globaldefs::NamingAttributesList_T** qosRuleNames,
out **HW_mstpInventory::HW_Flow_T** flow)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 删除指定的流。

编程指引：

void deleteFlow(in **globaldefs::NamingAttributes_T** flowName)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询所有的链路聚合组的名字。

编程指引：

void **getAllLinkAggregationGroupNames**(in **globaldefs::NamingAttributes_T** meName,
in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询所有的链路聚合组的信息。

编程指引：

void **getAllLinkAggregationGroups**(in **globaldefs::NamingAttributes_T** meName,
in unsigned long how_many,
out **HW_mstpInventory::HW_LinkAggregationGroupList_T** lagList,
out **HW_mstpInventory::HW_LinkAggregationGroupIterator_I** lagIt)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 创建链路聚合组。

编程指引：

void **createLinkAggregationGroup**(in **globaldefs::NamingAttributes_T** meName,
in **globaldefs::NVSList_T** paraList,
in **globaldefs::NamingAttributes_T** mainPortName,
in **globaldefs::NamingAttributesList_T** branchPortNameList,
out **HW_mstpInventory::HW_LinkAggregationGroup_T** lag)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询支持链路聚合功能的以太网单板端口信息。

编程指引：

void **getAvailablePortNames**(in **globaldefs::NamingAttributes_T** equipmentName,
in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定的链路聚合组信息。

编程指引：

void **getLinkAggregationGroup**(in **globaldefs::NamingAttributes_T** lagName,
out **HW_mstpInventory::HW_LinkAggregationGroup_T** lag)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 修改指定的链路聚合组。

编程指引：

void **modifyLinkAggregationGroup**(in **globaldefs::NamingAttributes_T** lagName,
in **globaldefs::NVSList_T** paraList,
in **globaldefs::NamingAttributesList_T** addedBranchPortNameList,
in **globaldefs::NamingAttributesList_T** deletedBranchPortNameList,
out **HW_mstpInventory::HW_LinkAggregationGroup_T** lag)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 删除指定的链路聚合组。

编程指引：

void **deleteLinkAggregationGroup**(in **globaldefs::NamingAttributes_T** lagName)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 设置指定端口的 Shaping 队列。

编程指引：

void **setMstpEndPointShapingQueue**(in **globaldefs::NamingAttributes_T** endPointName,
in **HW_mstpInventory::SharpingQueueList_T** sharpingQueueList)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定端口的 Sharping 队列。

编程指引：

void **getMstpEndPointShapingQueue**(in **globaldefs::NamingAttributes_T** endPointName,
out **HW_mstpInventory::SharpingQueueList_T** sharpingQueueList)
raises (globaldefs::ProcessingFailureException)

3.11.2 MSTP 业务管理

T2000 CORBA 接口提供对 MSTP 设备的业务管理功能，该部分功能参考 TMF814 建议，根据华为自定义 IDL 开发，实现对以太网专线（EPL）业务、以太网虚拟专线（EVPL）业务、以太网 LAN（EPLn）业务、以太网虚拟 LAN（EVPLn）业务以及 ATM　VPC/VCC 交叉连接的管理功能。

- 允许 NMS 查询指定网元内所有的以太网业务（包括 EPL/EVPL/EPLn/EVPLn）的特征信息。

编程指引：

void **getAllEthService**(in **globaldefs::NamingAttributes_T** meName,
in **HW_EthServiceTypeList_T** typeList,
in unsigned long how_many,
out **HW_EthServiceList_T** serviceList,
out **HW_EthServiceIterator_I** serviceIt)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定的以太网业务（包括 EPL/EVPL/EPLn/EVPLn）的特征信息。

编程指引：

void **getEthService**(in **globaldefs::NamingAttributes_T** serviceName,
out **HW_EthService_T** ethService)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 创建以太网业务（包括 EPL/EVPL/EPLn）。

编程指引：

void **createEthService**(in **HW_EthServiceCreateData_T** createData,
out **HW_EthServiceList_T** ethServiceList)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 删除指定的以太网业务。

编程指引：

void **deleteEthService**(in **globaldefs::NamingAttributes_T** serviceName)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定网元内所有的 ATM VP/VC 交叉连接的特征信息。

编程指引：

void **getAllAtmService**(in **globaldefs::NamingAttributes_T** meName,
in **HW_AtmServiceTypeList_T** typeList,
in unsigned long how_many,
out **HW_AtmServiceList_T** serviceList,
out **HW_AtmServiceIterator_I** serviceIt)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定的 ATM VP/VC 交叉连接的特征信息。

编程指引：

void **getAtmService**(in **globaldefs::NamingAttributes_T** serviceName,
out **HW_AtmService_T** atmService)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 创建 ATM VP/VC 交叉连接。

编程指引：

void **createAtmService**(in **HW_AtmServiceCreateData_T** createData,
out **HW_AtmService_T** atmService)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 删除指定的 ATM VP/VC 交叉连接。

编程指引：

void **deleteAtmService**(in **globaldefs::NamingAttributes_T** serviceName)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 激活指定的 ATM VP/VC 交叉连接。

编程指引：

void **activateAtmService**(in **globaldefs::NamingAttributes_T** serviceName,
out **HW_AtmService_T** atmService)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 去激活指定的 ATM VP/VC 交叉连接。

编程指引：

void **deactivateAtmService**(in **globaldefs::NamingAttributes_T** serviceName,
out **HW_AtmService_T** atmService)
raises(globaldefs::ProcessingFailureException)

3.11.3 MSTP 保护管理

T2000 CORBA 接口提供对 MSTP 设备的保护管理功能，该部分功能参考 TMF814 建议，根据华为自定义 IDL 开发，实现对以太网 RPR 保护以及 ATM VP/VC 保护的管理功能。

- 允许 NMS 查询网元内所有的 RPR 节点的特征信息。

编程指引：

void **getAllRPRNode**(in **globaldefs::NamingAttributes_T** meName,
in unsigned long how_many,
out **HW_RPRNodeList_T** nodeList,
out **HW_RPRNodeIterator_I** nodeIt)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定的 RPR 节点的特征信息。

编程指引：

void **getRPRNode**(in **globaldefs::NamingAttributes_T** nodeName,
out **HW_RPRNode_T** node)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 查询 RPR 节点的拓扑参数信息。

编程指引：

void **getRPRTopoPara**(in **globaldefs::NamingAttributes_T** nodeName,
out **HW_RPRTopoInfo_T** topoInfo)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 查询 RPR 节点的倒换状态。

编程指引：

void **retrieveRPRSwitchData**(in **globaldefs::NamingAttributes_T** nodeName
out **HW_RPRSwitchData_T** switchData)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 对指定的 RPR 节点执行外部倒换命令。

编程指引：

void **performRPRProtectionCommand(** in **globaldefs::NamingAttributes_T** nodeName,
in **protection::ProtectionCommand_T** protectionCommand,
in **HW_SwitchPosition_T** switchPosition,
out **HW_RPRSwitchData_T** switchData)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定网元内所有的 ATM 保护组的特征信息。

编程指引：

void **getAllAtmProtectGroup**(in **globaldefs::NamingAttributes_T** meName,
in unsigned long how_many,
out **HW_AtmProtectGroupList_T** atmPGList,
out **HW_AtmProtectGroupIterator_I** pgIt)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定的 ATM 保护组的特征信息。

编程指引：

void **getAtmProtectGroup**(in **globaldefs::NamingAttributes_T** atmpgName,
out **HW_AtmProtectGroup_T** atmPG)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定的 ATM 保护组的倒换状态。

编程指引：

void **retrieveAtmPGSwitchData**(in **globaldefs::NamingAttributes_T** atmpgName,
out **HW_AtmPGSwitchData_T** switchData)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 对指定的 ATM 保护组执行外部倒换命令。

编程指引：

void **performAtmPGProtectionCommand**(in **globaldefs::NamingAttributes_T** atmpgName,
in **protection::ProtectionCommand_T** protectionCommand,
in **HW_AtmPGSwitchAction_T** switchAction,
in **HW_AtmPGSwitchDirect_T** switchDirect,
out **HW_AtmPGSwitchData_T** switchData)
raises(globaldefs::ProcessingFailureException)

3.11.4 流量描述符管理

T2000 CORBA 接口提供对 MSTP 设备的流量描述符管理功能，该部分功能依据 TMF814 所定义的 IDL 接口（华为有部分扩展）开发，实现对 ATM 流量描述符的管理功能。

- 允许 NMS 查询网元内所有的 ATM 流量描述符的特征信息。

编程指引：

void **HW_getAllTrafficDescriptors** (in **globaldefs::NamingAttributes_T** meName,
in unsigned long how_many,
out **TrafficDescriptorList_T** trafficDescList,
out **TrafficDescriptorIterator_I** trafficDescIt)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询网元内所有的 ATM 流量描述符的名字。

编程指引：

void **HW_getAllTrafficDescriptorNames**(in **globaldefs::NamingAttributes_T** meName,
in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 激活指定的 ATM 流量描述符。

编程指引：

void **activateTrafficDescriptor**(in **globaldefs::NamingAttributes_T** tdName,
out T**rafficDescriptor_T** td)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 去激活指定的 ATM 流量描述符。

编程指引：

void **deactivateTrafficDescriptor**(in **globaldefs::NamingAttributes_T** tdName,
out **TrafficDescriptor_T** td)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 创建 ATM 流量描述符。

编程指引：

void **HW_createTrafficDescriptor**(in **globaldefs::NamingAttributes_T** meName,
in **TDCreateData_T** newTDCreateData,
out **TrafficDescriptor_T** newTrafficDescriptor)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 删除指定的 ATM 流量描述符。

编程指引：

void **deleteTrafficDescriptor**(in **globaldefs::NamingAttributes_T** descriptorName)
raises (globaldefs::ProcessingFailureException)

- 允许 NMS 查询指定流量描述符的信息。

编程指引：

void **getTrafficDescriptor**(in **globaldefs::NamingAttributes_T** tdName,
out **TrafficDescriptor_T** td)
raises (globaldefs::ProcessingFailureException)

3.11.5 封装层链路管理

- 供高层网管查询 EMS 内所有的 ELL 的名字

编程指引：

void **getAllELLinkNames**(in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises(globaldefs::ProcessingFailureException)

- 供高层网管查询 EMS 内所有的 ELL 的详细信息

编程指引：

void **getAllELLinks**(in unsigned long how_many,
out **ELLinkList_T** ells,
out **ELLinkIterator_I** ellIt)
raises(globaldefs::ProcessingFailureException)

- 根据给出的 ELL 名字查询其详细信息

编程指引：

void **getELLink**(in **globaldefs::NamingAttributes_T** ellName,
out **EncapsulationLayerLink_T** ell)
raises(globaldefs::ProcessingFailureException)

- 提供创建 ELL 的功能，目前支持自动和手工两种方式创建服务层路径

编程指引：

void **createELLink**(in **ELLinkCreateData_T** createData,
out **EncapsulationLayerLink_T** theELL,
out string errorReason)
raises (globaldefs::ProcessingFailureException)

- 激活指定的 ELL

编程指引：

void **activateELLink**(in **globaldefs::NamingAttributes_T** ellName,
out **EncapsulationLayerLink_T** theELL,
out string errorReason)
raises (globaldefs::ProcessingFailureException);

- 去激活指定的 ELL

编程指引：

void **deactivateELLink** (in **globaldefs::NamingAttributes_T** ellName,
out **EncapsulationLayerLink_T** theELL,
out string errorReason)
raises (globaldefs::ProcessingFailureException);

- 删除指定的 ELL

编程指引：

void **deleteELLink**(in **globaldefs::NamingAttributes_T** ellName)
raises (globaldefs::ProcessingFailureException);

- 增加指定的 ELL 的服务层带宽，对未终结业务只支持手工增加带宽

编程指引：

void **increaseBandwidthOfELLink**(in **globaldefs::NamingAttributes_T** ellName,
in boolean automatic,
in **SNCCreateDataList_T** additionalSNCs,
in short numberOfSNCs,
in **transmissionParameters::LayeredParameterList_T** transmissionParams,
in **globaldefs::NVSList_T** additionalModificationInfo,
out **EncapsulationLayerLink_T** newELL)
raises (globaldefs::ProcessingFailureException)

- 减少指定的 ELL 的服务层带宽

编程指引：

void **decreaseBandwidthOfELLink**(in **globaldefs::NamingAttributes_T** ellName,
in **globaldefs::NamingAttributesList_T** sncNames,
in short numberOfSNCs,
in **globaldefs::NVSList_T** additionalModificationInfo,
out **EncapsulationLayerLink_T** newELL)
raises (globaldefs::ProcessingFailureException)

- 设置 ELL 的 LCAS 状态

编程指引：

void **setELLinkLCASState** (in **globaldefs::NamingAttributes_T** ellName,
in boolean enableState)
raises (globaldefs::ProcessingFailureException)

3.11.6 流域管理

- 查询 EMS 内所有 FD 的详细信息

编程指引：

void **getAllFlowDomains**(in unsigned long how_many,
out **FDList_T** flowDomains,
out **FDIterator_I** fdIt)
raises(globaldefs::ProcessingFailureException)

- 查询 EMS 内所有 FD 的名字

编程指引：

void **getAllFlowDomainNames**(in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises(globaldefs::ProcessingFailureException)

- 根据 FD 的名字查询其详细信息

编程指引：

void **getFlowDomain**(in **globaldefs::NamingAttributes_T** fdName,
out **FlowDomain_T** flowDomain)
raises(globaldefs::ProcessingFailureException)

- 查询指定 FD 内所有的 FDFr 的详细信息

编程指引：

void **getAllFDFrs**(in **globaldefs::NamingAttributes_T** fdName,
in unsigned long how_many,
in **transmissionParameters::LayerRateList_T** connectivityRateList,
out **flowDomainFragment::FDFrList_T** fdfrList,
out **flowDomainFragment::FDFrIterator_I** fdfrIt)
raises(globaldefs::ProcessingFailureException)

- 查询指定 FD 内所有的 FDFr 的名字

编程指引：

void **getAllFDFrNames**(in **globaldefs::NamingAttributes_T** fdName,
in unsigned long how_many,
in **transmissionParameters::LayerRateList_T** connectivityRateList,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises(globaldefs::ProcessingFailureException)

- 根据 FDFr 的名字查询其详细信息

编程指引：

void **getFDFr(in globaldefs::NamingAttributes_T** fdfrName,
out **flowDomainFragment::FlowDomainFragment_T** fdfr)
raises (globaldefs::ProcessingFailureException)

- 创建以太网业务，目前支持创建单点和未终结的 EPL、EVPL，支持 EPLAN 业务与 QINQ 业务，不支持 EVPLAN

编程指引：

void **createFDFr**(in **flowDomainFragment::FDFrCreateData_T** createData,
in **ConnectivityRequirement_T** connectivityRequirement,
inout **globaldefs::NamingAttributesList_T** endTPs,
inout **globaldefs::NamingAttributesList_T** internalTPs,
inout **subnetworkConnection::TPDataList_T** tpsToModify,
out **flowDomainFragment::FlowDomainFragment_T** theFDFr,
out string errorReason)
raises (globaldefs::ProcessingFailureException);

- 激活指定的 FDFr

编程指引：

void **activateFDFr**(in **globaldefs::NamingAttributes_T** fdfrName,
out **flowDomainFragment::FlowDomainFragment_T** fdfr)
raises (globaldefs::ProcessingFailureException);

- 去激活指定的 FDFr

编程指引：

void **deactivateFDFr**(in **globaldefs::NamingAttributes_T** fdfrName,
out **flowDomainFragment::FlowDomainFragment_T** fdfr)
raises (globaldefs::ProcessingFailureException);

- 删除指定的 FDFr

编程指引：

void **deleteFDFr**(in **globaldefs::NamingAttributes_T** fdfrName,
inout **subnetworkConnection::TPDataList_T** tpsToModify)
raises (globaldefs::ProcessingFailureException);

- 查询指定 FDFr 的服务层路径

编程指引：

void **getFDFrServerTrail** (in **globaldefs::NamingAttributes_T** fdfrName,
out **globaldefs::NamingAttributesList_T** serverNameList)
raises (globaldefs::ProcessingFailureException);

- 查询 EMS 内所有的以太网 OAM 维护点信息

编程指引：

void **getAllEthernetOAMPoint** (in **globaldefs::NamingAttributes_T** Fdfrname,
out **EthernetOAMPointList_T** oamPointList)
raises (globaldefs::ProcessingFailureException);

- 执行以太网 OAM 维护命令

编程指引：

void **performEthernetOAMCommand** (in **EthernetOAMOperation_T** operation,
out **EthernetLTTestResultList_T** ltTestResult)
raises (globaldefs::ProcessingFailureException)

3.12 控制平面管理

- 允许 NMS 查询 T2000 管理域内所有的路由域。

编程指引：

void **getAllRoutingAreaNames**(out **globaldefs::NamingAttributesList_T** nameList)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 根据路由域的名称，查询路由域内所有路由节点的名称。

编程指引：

void **getAllRoutingNodeNames**(in **globaldefs::NamingAttributes_T** routingAreaName,
in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nodeNameList,
out **globaldefs::NamingAttributesIterator_I** nodeIt)
raises(globaldefs::ProcessingFailureException);

- 允许 NMS 根据路由域的名称，查询路由域内所有的 SNPPLink。

编程指引：

void **getAllSnppLinks**(in **globaldefs::NamingAttributes_T** routingAreaName,
in unsigned long how_many,
out **HW_SnppLinkList_T** snppLinkList,
out **HW_SnppLinkIterator_I** snppLinkIt)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 根据智能节点的名称，查询该智能节点内包含的所有 SNPP 的名称。

编程指引：

void **getAllSnppNames**(in **globaldefs::NamingAttributes_T** routingNodeName,
in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 根据 SNPP 的名称，查询 SNPP 中包含的所有 SNP 的名称。

编程指引：

void **getAllContainedSnpNames**(in globaldefs::NamingAttributes_T snppName,
in unsigned long how_many,
out **globaldefs::NamingAttributesList_T** nameList,
out **globaldefs::NamingAttributesIterator_I** nameIt)
raises(globaldefs::ProcessingFailureException)

- 允许 NMS 设置 SNPPLink 的共享链路风险组属性。

编程指引：

void **setSRLG**(in **HW_SRLGID_T** srlgID,
in boolean addOrRemove,
inout **globaldefs::NamingAttributesList_T** snppLinkNameList)
raises(globaldefs::ProcessingFailureException)

3.13 界面直通

为使 NMS 更加直观、更加方便地实施对 T2000 的管理功能，T2000 CORBA 接口提供了 GCT（Graphic User Interface Cut-Through）功能。即可以实现 T2000-NMS 间的界面同步显示功能。NMS 借助此功能，可以直接在本地机器上访问到 T2000 的图形界面，从而实现对 T2000 的直接操作与管理功能。

TMF MTNM 规定了两种界面直通方式：Client launch 方式和 Server launch 方式，T2000 CORBA 接口对这两种方式均支持。

- 允许 NMS 查询界面直通信息，该信息包含：GCT Server 的地址、client 方式启动的命令、gctScope、操作系统类型、gctContext 等。

编程指引：

void **getGCTProfileInfo**(out **GCTProfileInfo_T** gctProfileInfo)
raises (globaldefs::ProcessingFailureException)

- 启动 GCT：允许 NMS 在指定 IP 上启动 GCT。

编程指引：

void **launchGCT**(in **globaldefs::NamingAttributes_T** objectName,
in string gctContext,
in **globaldefs::NVSList_T** userInfo,
in string displayAddress,
in **globaldefs::NVSList_T** additionalInputInfo,
out boolean closingEnabled,
out **globaldefs::NVSList_T** additionalOutputInfo)
raises(globaldefs::ProcessingFailureException)

- 关闭 GCT：允许 NMS 在指定 IP 上关闭所有 GCT。

编程指引：

void **destroyGCT**(in string displayAddress)
raises(globaldefs::ProcessingFailureException)

3.14 通知上报

在子网资源被更新时（例如：链路的增删，网元的增删等），借助 CORBA Notification Service，T2000 可以及时将存量的变更情况通知 NMS。使得 NMS 掌握最新的网络动态，确保与 T2000 的数据一致。T2000 CORBA 接口支持以下通知功能。

- 告警上报通知(NT_ALARM)

  每条告警的开始和结束，EMS 都会向 NMS 上报一条告警事件。
- 性能越限告警通知(NT_TCA)

  当前性能值超过性能高门限时，EMS 会向 NMS 上报 TCA 事件。
- 文件传输状态通知(NT_ FILE_TRANSFER_STATUS)

  当 NMS 查询 EMS 上的历史性能数据时，EMS 通过 FTP 以文本方式将历史性能数据传送给 NMS，在这个过程中 EMS 会上报文件传送状态通知，反馈文件传输的当前状态和完成进度。
- 对象创建通知(NT_OBJECT_CREATION)

  支持 NMS 向 T2000 订阅下列对象的创建通知：
  - 设备（支持单板创建通知，不支持槽位创建通知）
  - 子架（仅支持 DWDM 设备的子架创建通知）
  - 网元
  - PTP
  - Topological Link
  - SNC
  - Protection Group（包括 SDH 保护组，波分保护组，ATM 保护组，RPR 保护组，设备保护组）
  - VB
  - VLAN
  - MSTPEndPoint
  - QoS

- FLOW
- LAG
- 太网业务
- ATM 业务
- RPR 节点
- ATM 流量描述
- SNPPLink
- FDFr
- ELL
- 对象删除通知(NT_OBJECT_DELETION)
- 支持 NMS 向 T2000 订阅下列对象的删除通知：
- 设备（支持单板创建通知，不支持槽位创建通知）
- 子架（仅支持 DWDM 设备的子架创建通知）
- 网元
- Topological Link
- SNC
- Protection Group（包括 SDH 保护组，波分保护组，ATM 保护组，RPR 保护组，设备保护组）
- VB
- VLAN
- MSTPEndPoint
- QoS
- FLOW
- LAG
- 太网业务
- ATM 业务
- RPR 节点
- ATM 流量描述
- SNPPLink
- FDFr
- ELL

- 属性改变通知(NT_ATTRIBUTE_VALUE_CHANGE)

当某些对象的一个或多个属性改变时，T2000 会发送相应的属性值改变通知，如表 3-1 所示。

表3-1 支持上报的属性

对象类型	支持上报的属性
EMS	userlabel、owner
PTP（SDH、WDM）	userlabel、owner、 edgePoint、transmissionParams

对象类型	支持上报的属性
Equipment	userlabel、owner
SNC	userlabel、owner、 aEnd、zEnd
TopologicalLink	userlabel、owner
PGP	userlabel、owner
MultiLayerSubnetwork	userlabel、owner
ME	userlabel、owner、supportedRates
VB	userlabel、owner
VLAN	userlabel、owner
MSTPEndPoint	userlabel、owner
QoS	userlabel、owner
FLOW	userlabel、owner
LAG	userlabel、owner
ELL 通道绑定	BindingPath
ATM 保护组	userlabel、owner
RPR 节点	userlabel、owner
SNPPLink	userlabel、owner
FDFr	userlabel、owner
ELL	userlabel、owner

- 状态改变通知(NT_STATE_CHANGE)

当某些对象的状态改变时，T2000 会发送相应的状态改变通知，如表 3-2 所示。

表3-2 支持上报的状态

对象类型	支持上报的状态
ME	communicationState、emsInSyncState
SNC	sncState
PGP（SDH）	protectionSchemeState
LCAS	LCASEnableState
ATM 业务激活/去激活	activeState

对象类型	支持上报的状态
ATM 保护组状态	useState
ATM 流量描述激活/去激活	activeState
FDFr 激活/去激活	ActivateStatus
ELL 激活/去激活	ActivateStatus

- SDH 保护组倒换通知(NT_PROTECTION_SWITCH)

  当 SDH 保护组发生倒换时，T2000 会上报相应的保护组倒换通知。
- ATM 保护组倒换通知(NT_ATMPROTECTION_SWITCH)

  当 ATM 保护组发生倒换时，T2000 会上报相应的保护组倒换通知。
- 波分保护组倒换通知(NT_WDMPROTECTION_SWITCH)

  当波分保护组发生倒换时，T2000 会上报相应的保护组倒换通知。
- RPR 保护组倒换通知(NT_RPRPROTECTION_SWITCH)

  当 RPR 环网发生倒换时，T2000 会上报相应的保护倒换通知。
- 设备保护组倒换通知(NT_EPROTECTION_SWITCH)

  当设备保护组发生倒换时，如交叉板，时钟板的倒换，T2000 会上报相应的保护组倒换通知。
- 路由改变通知(NT_ROUTE_CHANGE)

  当 SNC 静态路由发生改变时，T2000 会上报路由改变事件，通知 NMS 改变后的路由详细信息。
- 智能域资源改变通知(NT_ASON_RESOURCE_CHANGE)

  创建智能域、删除智能域、创建智能路由节点、删除智能路由节点时，T2000 会给 NMS 上报该通知事件。
- 2M 伪随机码测试状态通知(NT_PRBSTEST_STATUS)

  对 4.0 和 5.0 平台设备进行 2M 伪随机码测试时，T2000 会向 NMS 上报该事件，通知测试的当前状态和完成进度。
- 心跳连接通知(NT_HEARTBEAT)

  当 NMS 和 T2000 通讯连接状态正常时，T2000 会定时上报该事件，表示 NMS 和 EMS 之间的通讯状态正常。

3.15 安全管理

- 安全验证

在允许 NMS 访问 T2000 之前，安全验证功能需要对 NMS 用户身份的合法性进行校验。

- Session 管理
  - 在合法性验证通过以后，允许 NMS 和 T2000 间建立 Session 连接
  - 允许同一个 NMS 和 T2000 间建立多个 Session 连接。

编程指引：

void **getEmsSession**(in string user,
in string password,
in **nmsSession::NmsSession_I** client,
out **emsSession::EmsSession_I** emsSessionInterface)
raises(globaldefs::ProcessingFailureException)

- NMS-T2000 通信状态监视

支持 NMS 和 T2000 间通信连接状态的定时检测，如果通信状态出现异常，T2000 侧会自动中止连接并释放相关资源。

编程指引：

void **ping** ()

3.16 异常管理

T2000 CORBA 接口提供了完善的 Session 管理功能。能自动检测出 NMS Session 的正常或异常终止，并及时正确地清理 T2000 中所有为该 NMS Session 生成的各种资源。保证了系统的安全性和可靠性，有效防止内存泄漏，避免服务器内存资源不足。

4 系统配置与组网应用

关于本章

本章描述内容如下表所示。

标题	内容
4.1 硬件配置	T2000 CORBA 接口硬件配置介绍。
4.2 软件配置	T2000 CORBA 接口软件配置介绍。
4.3 组网方式	T2000 CORBA 接口组网方式介绍。
4.4 高可用性系统配置	T2000 CORBA 接口高可用性系统配置介绍。

4.1 硬件配置

T2000 CORBA 接口和 T2000 Server 同机运行，要求机器配置能够安装运行 T2000 Server，不需要额外的硬件配置。

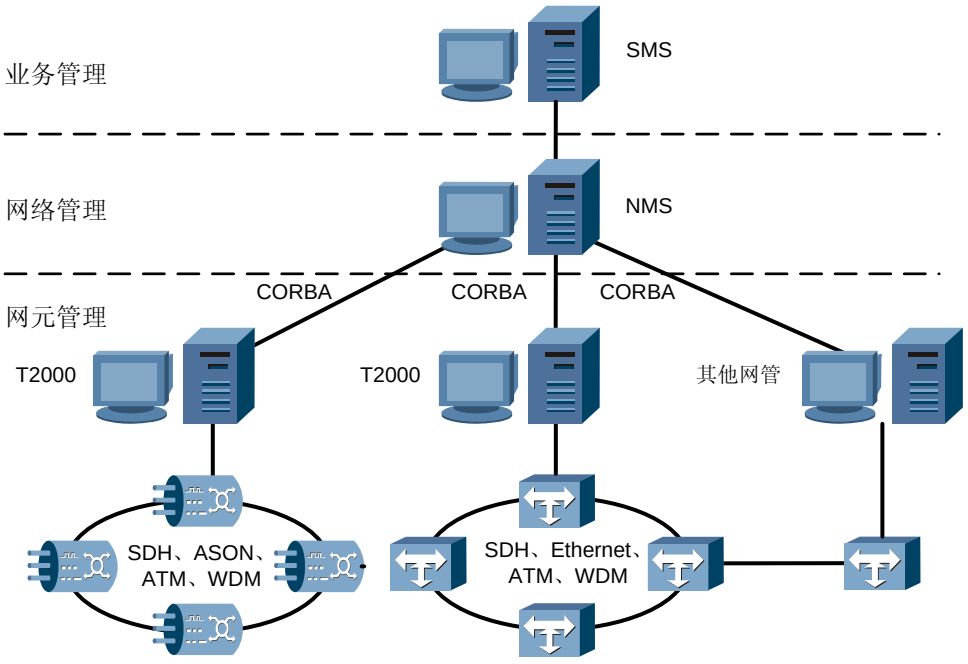
4.2 软件配置

T2000 CORBA 接口基于 TAO1.4.7（THE ACE ORB）的 ORB 技术实现。由于 TAO1.4.7 已经集成在 T2000 安装软件中。所以安装 T2000 CORBA 接口不需要额外的软件配置。

4.3 组网方式

T2000 CORBA 接口在综合网管中的组网方式如图 4-1 所示。

图4-1 T2000 CORBA 接口在网管体系中的组网方式



T2000 CORBA 接口作为 T2000 系统的一个部件，与 T2000 系统一起运行。

NMS 系统可以通过每个 T2000 的 CORBA 接口同时管理多个 T2000 系统，一个 T2000 系统也可以通过自身 CORBA 接口同时被多个 NMS 系统管理。当 T2000 存量发生改变时，T2000 CORBA 接口会通过事件通知上报给所有上层网管系统。

T2000 CORBA 提供的名字服务可以与 T2000 Server 集中部署在一起，也可以分开部署在高层网管 NMS 侧或任何一台机器。如图 4-2、图 4-3 所示。

图4-2 T2000 CORBA 接口服务的集中部署方式

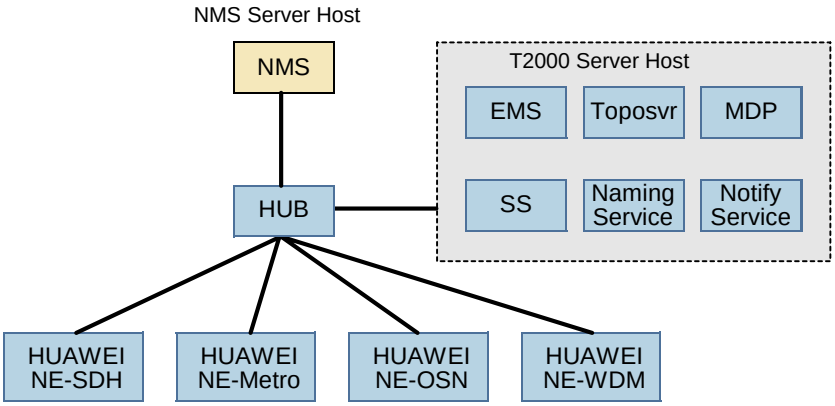
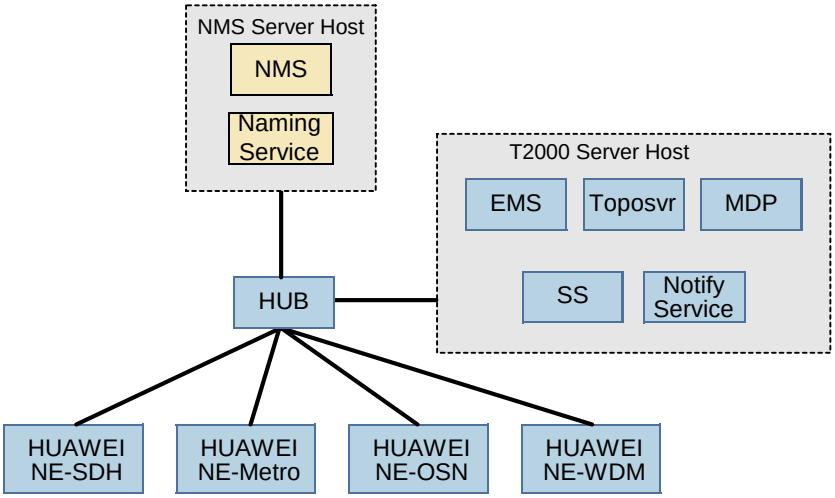


图4-3 T2000 CORBA 接口服务的分开部署方式



4.4 高可用性系统配置

4.4.1 Watchman 高可用性系统

概述

T2000 高可用性系统（Watchman）是 T2000 提供的高可用性系统中的一种，主要用于异地双机热备份。

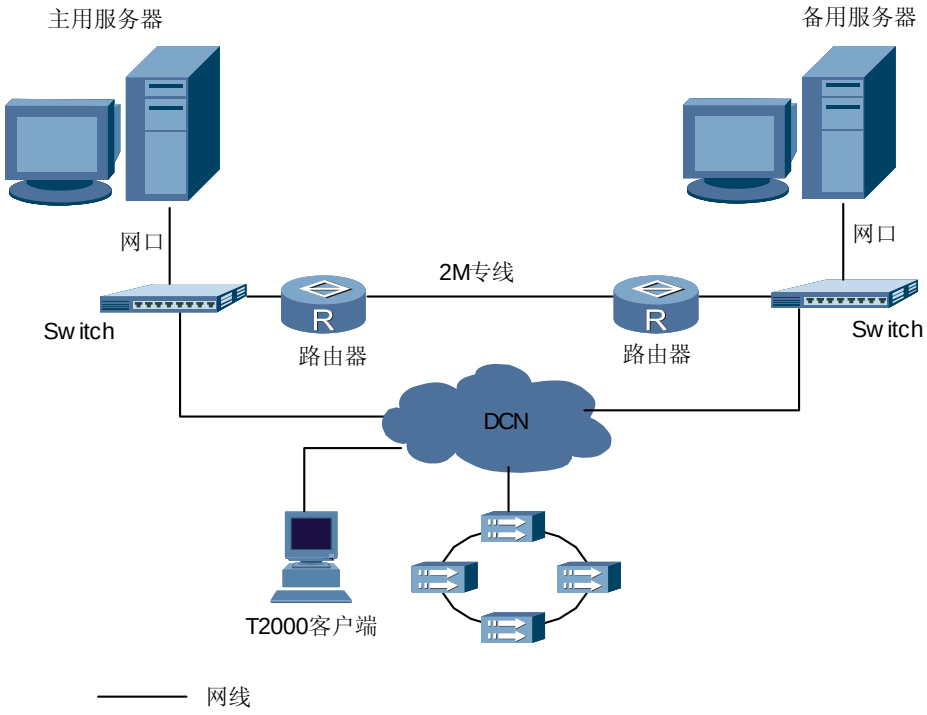
T2000 高可用性系统（Watchman）是一种高可靠性的传送网络管理系统。它在 T2000 的基础上，采用双机系统构成 T2000 服务器，Watchman 在线监控 T2000 服务器程序和数据复制服务的运行情况，使主备节点间保持通信，并通过卷复制软件实现数据备

份。当主节点出现故障时，可自动倒换到备节点，保证 T2000 对传送网络的不间断监控。

硬件连接图

T2000 高可用性系统（Watchman）的硬件连接图如图 4-4 所示。

图4-4 硬件结构图

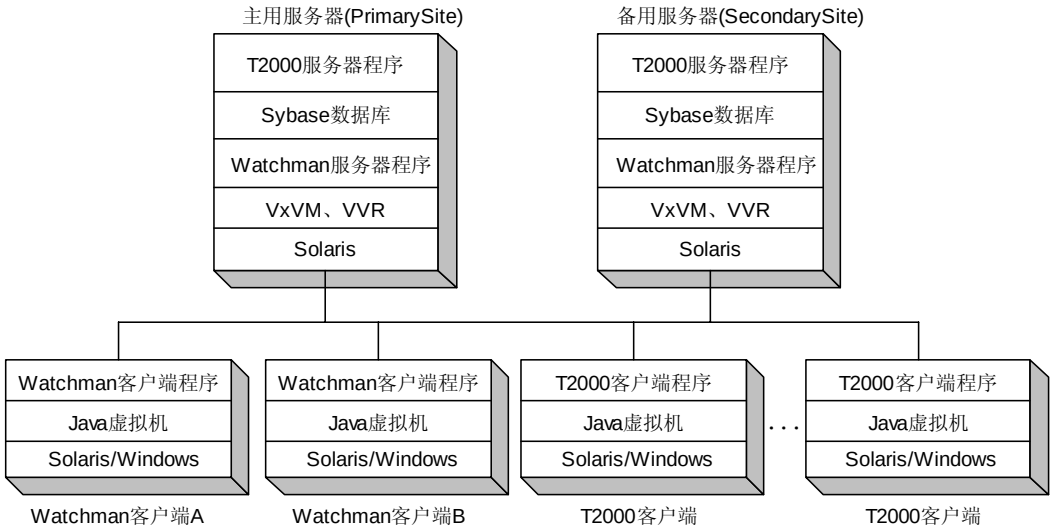


通常情况下，T2000 高可用性系统（Watchman）客户端、T2000 客户端软件都安装在独立的工作站或 PC 上。实际使用中，也可以将 T2000 高可用性系统（Watchman）客户端软件安装在主用和备用服务器上。

软件结构图

T2000 高可用性系统（Watchman）的软件结构图如图 4-5 所示。

图4-5 软件结构图



VVR	VERITAS Volume Replicator	VERITAS 卷复制器
VxVM	VERITAS Volume Manager	VERITAS 卷管理器

配置要求

说明

T2000 高可用性系统（Watchman）所需的软硬件运行环境经过了严格选型和测试，请遵循下列配置要求。

T2000 Server 和 CORBA 接口服务端的配置要求如表 1-1 所示。

表4-1 配置要求-高可用性系统（Watchman）服务器端

描述	数量	备注
SUN-Netra 240	2	必配。从两种方案中选择一种。
SUN Fire V890	2	
操作系统-Solaris-10	2	必配
数据库-Sybase Adaptive Server Enterprise -12.5	2	必配
软件套件-iManager T2000-中文版 UNIX 网管系统软件套件	2	必配
终端物理软件-iMap-Watchman 软件 (V100R002C10B213SP05)	2	必配
工具软件-VERITAS 4.1 异地备份软件-(含 VVM,VVR)	2	必配
iManager T2000-服务器 License 授权组件	2	必配

4.4.2 Veritas 温备份高可用性系统

概述

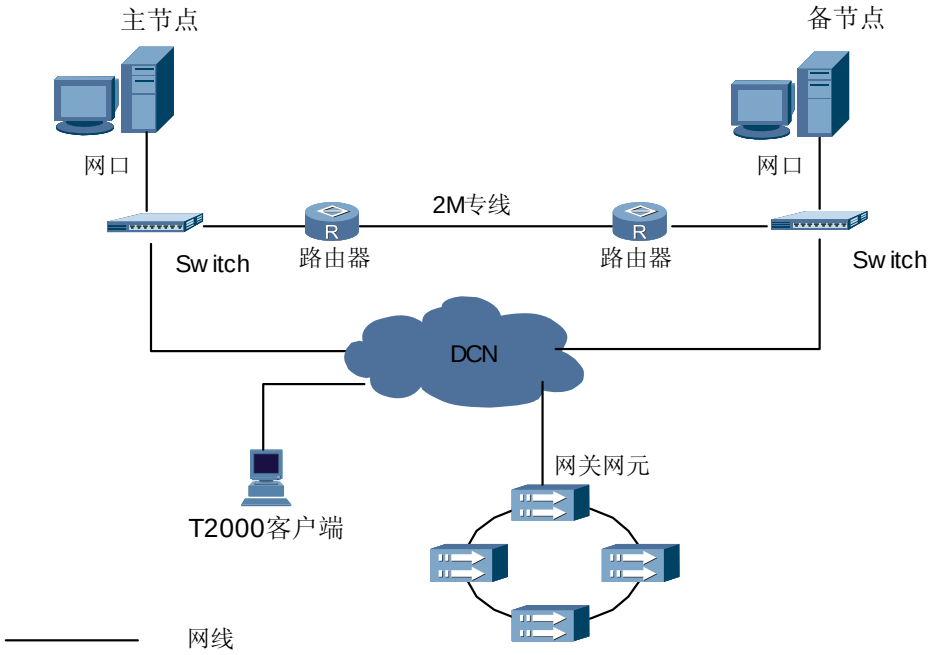
T2000 高可用性系统（Veritas）是 T2000 提供的高可用性系统解决方案中的一种，主要用于异地双机温备份。

T2000 高可用性系统（Veritas）是一种高可靠性的传送网络管理系统。它在 T2000 的基础上，采用双机系统构成 T2000 服务器，集成 Veritas 远程温备份技术，实现数据的实时备份，并动态监视 T2000 的运行状态。当主用服务器出现故障时，可手工倒换到备用服务器，保证 T2000 对传送网络的不间断监控。

硬件连接图

T2000 高可用性系统（Veritas）的硬件连接图如图 4-6 所示。

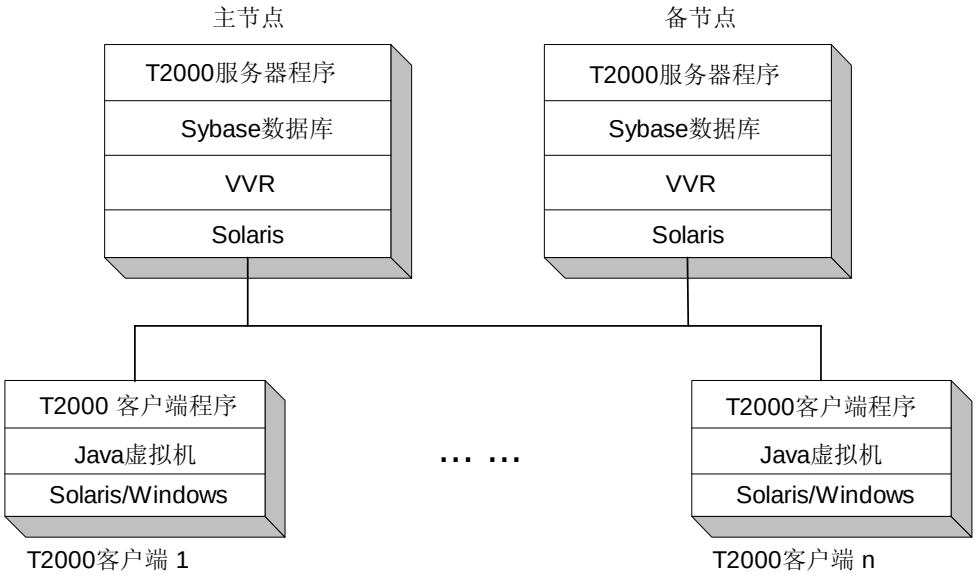
图4-6 硬件结构图



软件结构图

T2000 高可用性系统（Veritas）的软件结构图如图 4-7 所示。

图4-7 软件结构图



配置要求

T2000 Server 和 CORBA 接口服务端的高可用性系统（Veritas）中，主节点和备节点的配置相同。

表4-2 配置要求-高可用性系统（Veritas）服务器端

描述	数量	备注
SUN Fire V890	2	必配。从中选择一种。
SUN Netra V240	2	
操作系统-Solaris-10 (03/05)-多语言安装介质-标准版	2	必配
数据库-Sybase Adaptive Server-12.5	2	必配
软件套件-iManager T2000-中文版 UNIX 网管系统软件套件	2	必配
组件/附件-iManager T2000-服务器 License 授权组件	2	必配
工具软件-VERITAS-4.1 异地备份软件-(含 VVR)	2	必配

4.4.3 Veritas 热备份高可用性系统

概述

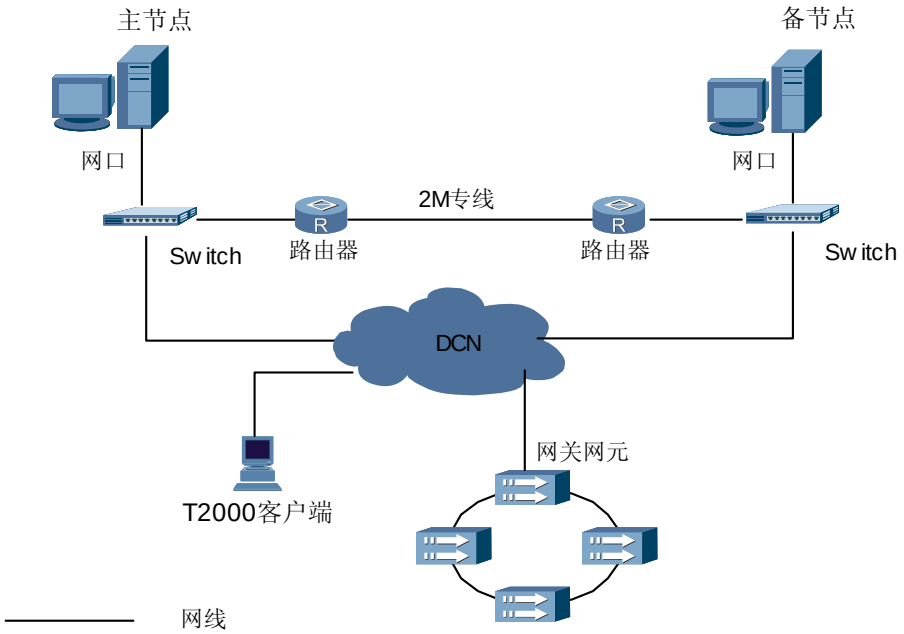
T2000 高可用性系统（Veritas）是 T2000 提供的高可用性系统解决方案中的一种，主要用于异地双机热备份。

T2000 高可用性系统（Veritas）是一种高可靠性的传送网络管理系统。它在 T2000 的基础上，采用双机系统构成 T2000 服务器，集成 Veritas 远程热备份技术，实现数据的实时备份，并动态监视 T2000 的运行状态。当主节点出现故障时，系统可自动倒换到备节点，保证 T2000 对传送网络的不间断监控。

硬件连接图

T2000 高可用性系统（Veritas）的硬件连接图如图 4-8 所示。

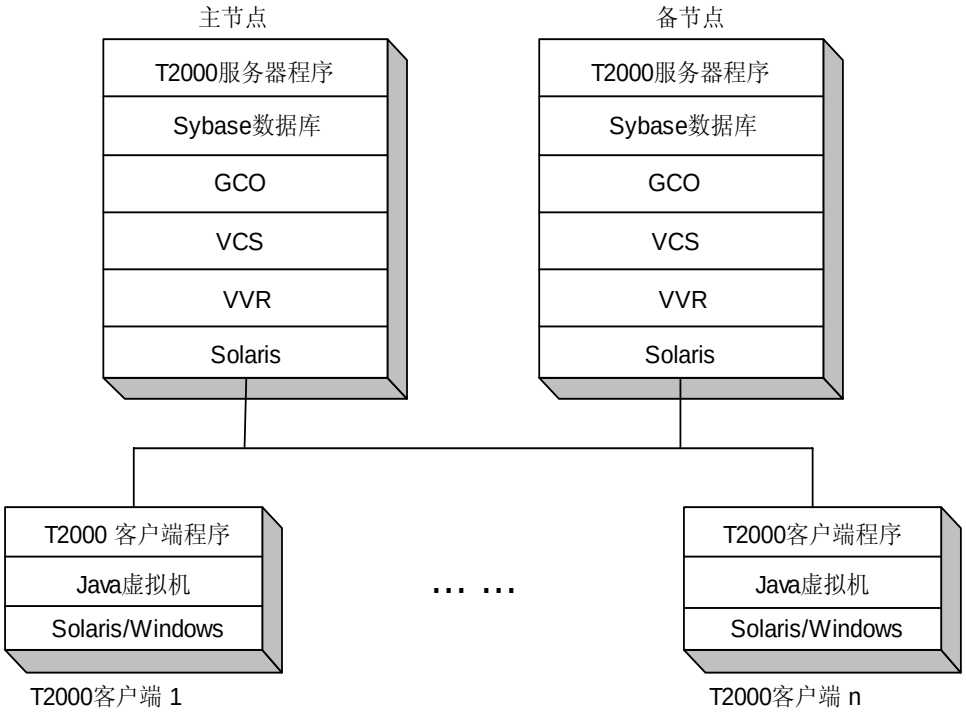
图4-8 硬件结构图



软件结构图

T2000 高可用性系统（Veritas）的软件结构图如图 4-9 所示。

图4-9 软件结构图



- GCO（Global Cluster Option）：VERITAS 远程集群管理器，在线监控系统和应用服务，当主用服务器的 T2000 服务器程序不可用的时候立即启动备用服务器上的 T2000 服务器程序。
- VCS：VERITAS 本地集群软件，实时监控系统和应用服务，当硬件或软件发生故障的时候，VCS 重新启动或者关闭应用服务。

配置要求

T2000 Server 和 CORBA 接口服务端的高可用性系统（Veritas）中，主节点和备节点的配置相同。

表4-3 配置要求-高可用性系统（Veritas）服务器端

描述	数量	备注
SUN Fire V890	2	必配。从中选择一种。
SUN Netra V240	2	
操作系统-Solaris-10 (03/05)-多语言安装介质-标准版	2	必配
数据库-Sybase Adaptive Server-12.5	2	必配
软件套件-iManager T2000-中文版 UNIX 网管系统软件套件	2	必配
组件/附件-iManager T2000-服务器 License 授权组件	2	必配
工具软件-VERITAS-4.1 异地备份软件-(含 VVR,VCS,GCO)	2	必配

4.4.4 Sun Cluster 高可用性系统

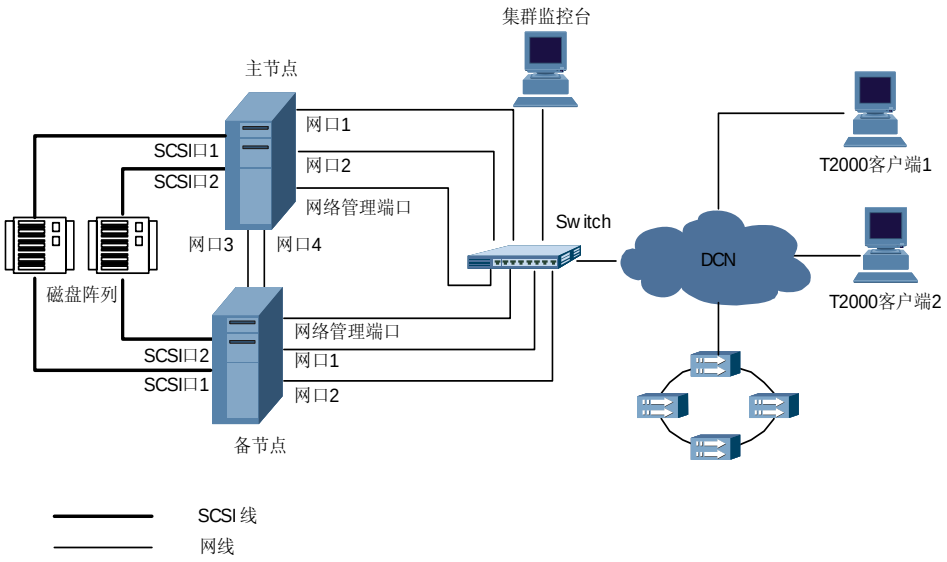
概述

T2000 高可用性系统（Sun Cluster）是 T2000 提供的高可用性系统中的一种，主要用于本地双机热备份。

硬件连接图

T2000 高可用性系统（Sun Cluster）的硬件连接图如图 4-10 所示。

图4-10 硬件连接图-高可用性系统（Sun Cluster）

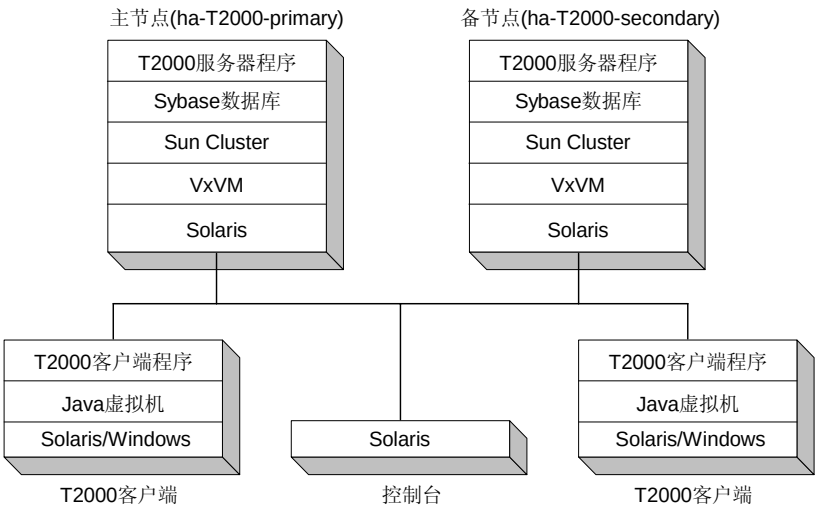


- 两台主机都安装了 T2000 服务器端软件，组成 Cluster 的两个节点，互为保护。
- 两个磁盘阵列互为镜像，并保持两台主机之间的数据同步。
- 两台主机之间使用两根交叉网线连接，采用私网的 IP 地址。
- 集群监控台（Cluster console）提供集群系统的单点管理，处理集群范围内的故障管理。
- 网络交换机（Switch）：用于网络数据交换。

软件结构图

T2000 高可用性系统（Sun Cluster）的软件结构图如图 4-11 所示。

图4-11 软件结构图-高可用性系统（Sun Cluster）



配置要求

T2000 Server 和 CORBA 接口服务端的高可用性系统（Sun Cluster）配置如表 4-4 所示。

表4-4 配置要求-高可用性系统（Sun Cluster）服务器端

描述	数量	备注
Netra240	2	必配
StorEdge3320 阵列	1	必配
软件套件-iManager T2000-中文版 UNIX 网管系统软件套件	2	必配
组件/附件-iManager T2000-服务器 License 授权组件	2	必配
操作系统-Sun Solaris-10-多语言安装介质	2	必配
数据库-Sybase Adaptive Server-12.5	2	必配
应用软件-Sun Cluster-3.1-Update 4 08/05-with Volume Manager 4.1-英文资料-	1	必配

高可用性系统（Sun Cluster）控制台配置如表 4-5 所示。

表4-5 配置要求-高可用性系统（Sun Cluster）控制台

描述	数量	备注
Blade 150	1	必配
操作系统-Sun Solaris-10-多语言安装介质-标准版-简化版英文资料	1	必配

5 系统安装和运行

关于本章

本章描述内容如下表所示。

标题	内容
5.1 检查与准备工作	介绍启动北向接口前的检查与准备工作
5.2 启动和禁用 CORBA 接口(单机)	T2000 CORBA 接口单机环境启动和停止介绍。
5.3 启动和禁用 CORBA 接口(双机)	T2000 CORBA 接口双机环境启动和停止介绍。
5.4 附录	附录

5.1 检查与准备工作

T2000 Server 的安装过程已经集成了 CORBA 接口软件的安装，因此无需额外进行 CORBA 接口软件的安装，但在 CORBA 接口软件正常使用前必须对其进行相关的软件参数配置，为了简化配置操作，推荐使用随 T2000 网管软件一起发布的图形化的 T2000 配置工具软件进行相关参数配置。

此外，用户若需使能 CORBA 接口功能，需要购买相应的 License。因此为确保后续的软件配置操作有效，请在进行 CORBA 接口软件配置之前，按照如下步骤完成相应的文件检查与准备工作：

- 检查 T2000 License
- 检查 T2000 配置工具
- 检查通知服务

5.1.1 检查 T2000 License

使用文本阅读工具软件查看$IMAP/licenseXXXXXXX.txt（Solaris 下使用 t2000 用户查看 License 文件，Windows 下使用 Administrator 用户查看 License 文件。其中 X 代表 0-9 间的整数）文件内容，如果 License 中 SupportCorbaIFBase=1，则 T2000 可以提供 CORBA 的基础功能，如果 Licnese 中 SupportCorbaIFBase=1 并且 SupportCorbaIFAdv=1，则 T2000 可以提供 CORBA 的基础和高级功能。T2000 License 文件中的控制项与 CORBA 功能项的关系请参见 ***“5.4.1 License 项与 CORBA 的关系”*** 小节。确认现有 License 支持 CORBA 功能。

说明

Solaris 环境中，环境变量表现形式为$Variable(Variable 为变量名称)。

Windows 环境中，环境变量表现形式为%Variable%(Variable 为变量名称)

$IMAP 环境变量指向 T2000 Server 的安装目录。

- 在 Solaris 环境中，T2000 Server 默认安装在如下目录：/T2000/server，则$IMAP 指向 /T2000/server。
- 在 Windows 环境中，T2000 Server 默认安装在 c:\T2000\server，则%IMAP%指向 c:\T2000\server。

后续环境变量均使用 Solaris 下的表达方式$IMAP。

5.1.2 检查 T2000 配置工具

由于 T2000 Server 的安装过程集成了 T2000 配置工具的安装，故无需额外进行配置工具的安装。

- 在 Solaris 下，T2000 配置工具安装在$IMAP/tools/configtool 目录下。
- 在 Windows 下，T2000 配置工具安装在%IMAP%\tools\configtool 目录下。

5.1.3 检查通知服务

T2000 网管系统提供了两种通知服务供用户进行选择，一种是免费中间件(TAO)，另一种是商业付费的中间件(IONA 的 orbix)。

- 使用 TAO 通知服务的 T2000 网管软件，支持单机系统和双机系统。
- 使用 IONA orbix 通知服务的 T2000 网管软件，仅支持单机系统。
- IONA 的 orbix 通知服务较 TAO 通知服务对可靠性事件有较好的支持。
- IONA 的 oribx 通知服务不支持 SSL 协议。

用户可以根据需要对这两种通知服务进行选择。T2000 网管系统默认提供的是 TAO 的通知服务。

使用 TAO 通知服务

如果用户希望使用 TAO 通知服务，需要按照如下步骤进行检查：

步骤 1 进入配置文件所在目录。

Solaris 下执行如下命令进入目录：

cd $IMAP/tools/configtool/config

Windows 下直接进入%IMAP%\tools\configtool\config 即可。

步骤 2 检查 config_item.properties 文件。

- 如果 CONFIG_TI_OR_CORBA_INTERFACE=0 则无需对文件进行修改。
- 如果 CONFIG_TI_OR_CORBA_INTERFACE=1 或 CONFIG_TI_OR_CORBA_INTERFACE=2，则需要修改为 CONFIG_TI_OR_CORBA_INTERFACE=0。保存文件并退出。

----结束

使用 IONA 通知服务

如果用户希望使用 IONA 通知服务，需要按照如下步骤进行检查：

步骤 1 申请或购买 orbix 的 license 文件，将申请后的 License 文件命名为“licenses.txt”然后将该文件放到$IMAP /orbix/etc/下。

步骤 2 Solaris 下使用 t2000 用户来确认以下环境变量已经设置并且正确。Windows 下使用 Administrator 用户来确认以下环境变量已经设置并且正确。

1. $IT_PRODUCT_DIR 变量已经设置，$IT_PRODUCT_DIR 指向$IMAP/orbix 目录。
2. $IT_DOMAIN_NAME 变量已经设置，$IT_DOMAIN_NAME 取值为”T2000”。
3. $IT_LICENSE_FILE 变量已经设置，$IT_LICENSE_FILE 指向文件 $IMAP/orbix/etc/licenses.txt。
4. $IT_CONFIG_DOMAINS_DIR 变量已经设置，$IT_CONFIG_DOMAINS_DIR 指向 $IMAP/orbix/etc/domains 目录。

注意

查看环境变量方法:

- Solaris 下查看环境变量的命令为:

echo $variable(variable 为环境变量名称)

- Windows 下查看环境变量的方法为:

右击"我的电脑" --> 属性 --> 高级 --> 环境变量，在系统变量中查看以上各个变量即可。

新建或设置环境变量:

- 在 Solaris 下，如果环境变量不存在或配置错误，需执行如下命令(使用 t2000 用户，举 IT_PRODUCT_DIR 为例):

IT_PRODUCT_DIR=$IMAP/orbix;export IT_PRODUCT_DIR

- 在 Windows 下，如果环境变量不存在，需执行如下操作(使用 Administrator 用户，举 IT_PRODUCT_DIR 为例):

右击"我的电脑" --> 属性--> 高级 --> 环境变量 在"系统变量" --> 新建 --> 在变量名中输入 *IT_PRODUCT_DIR*，在变量值中输入 *%IMAP%\orbix* --> 确认。

- 在 Windows 下，如果环境变量配置错误，需执行如下操作(使用 Administrator 用户，举 IT_PRODUCT_DIR 为例):

右击"我的电脑" --> 属性--> 高级 --> 环境变量 在"系统变量" 中选择" IT_PRODUCT_DIR " --> 编辑 在"变量值"一栏中输入: *%IMAP%\orbix* --> 确定。

步骤 3 检查$IMAP/tools/configtool/config/config_item.properties 文件，

如果 CONFIG_TI_OR_CORBA_INTERFACE=2 则无需对文件进行修改。

如果 CONFIG_TI_OR_CORBA_INTERFACE=0 或 CONFIG_TI_OR_CORBA_INTERFACE=1，则需要修改为 CONFIG_TI_OR_CORBA_INTERFACE=2。保存文件并退出。

----结束

5.2 启动和禁用 CORBA 接口(单机)

在 Solaris 环境下与 Windows 环境下，启用和禁用 CORBA 接口的步骤相同，以下描述中以 Solaris 环境为例。

注意

- 工作站环境下需要使用 t2000 用户登录 Solaris 系统
- PC 环境下需要使用 Administrator 用户登录 Windows 系统

说明

Solaris 和 Windows 环境下 T2000 Server 的目录结构是一致的。

5.2.1 启动 T2000 CORBA 接口

启动 T2000 CORBA 接口需进行如下操作：

- 使用 T2000 配置工具对 CORBA 接口进行配置
- 启动 T2000 网管系统

使用 T2000 配置工具对 CORBA 接口进行配置

配置工具可以自动识别 T2000 的版本、语言。不需要做任何配置，直接运行$IMAP/tools/configtool/configtool.sh(Windows 下运行 *configtool.bat*)即可使用配置工具。建议按照以下步骤使用配置工具配置 CORBA。

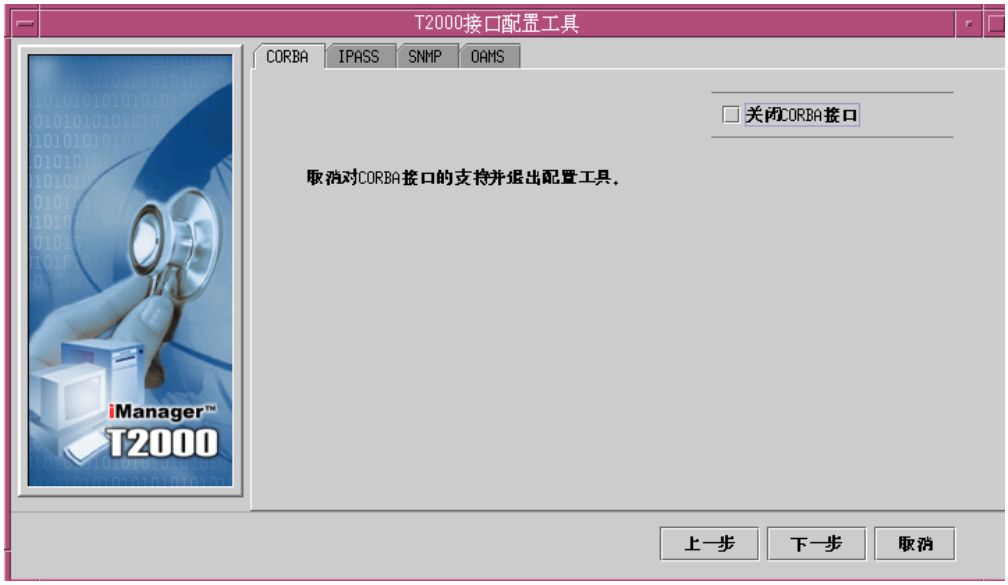
步骤 1 启动配置工具。

Solaris 下使用 t2000 用户执行命令来启动工具。

```
sh configtool.sh
```

PC 下使用 Administrator 用户直接双击 *configtool.bat* 来启动工具。

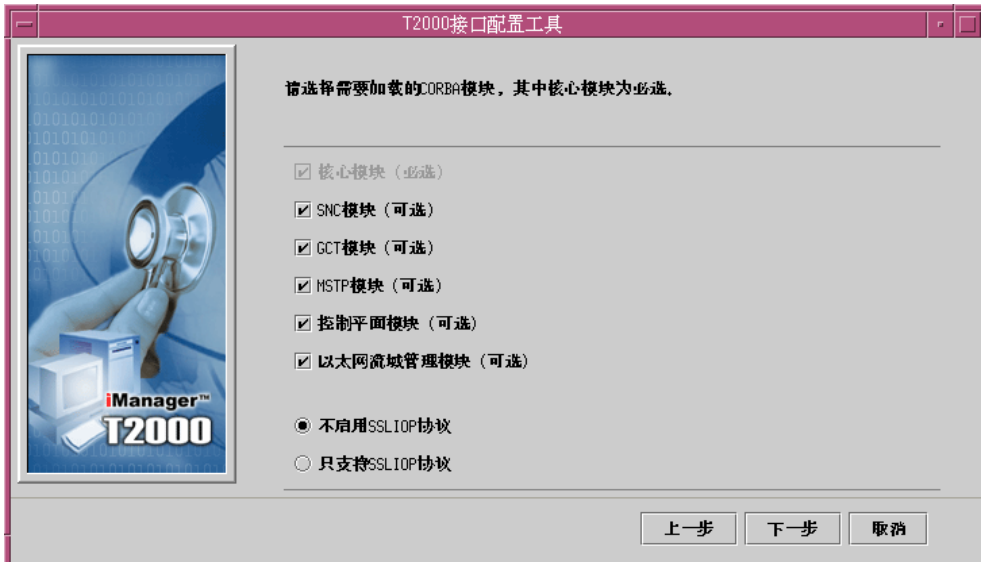
工具启动后，单击<下一步>。



步骤 2 单击<下一步>。

说明

如果选中"关闭 CORBA 接口"功能，单击<下一步>将退出 T2000 配置工具。



步骤 3 配置需要加载的 CORBA 模块，单击<下一步>。

说明

- 这里可以选择要加载的 CORBA 模块。其中"核心模块"为必选模块，其余模块请根据已有 license 支持情况选择加载。
- 对于 T2000V2R3C01 及以后版本，会出现“不启用 SSLIOP 协议”和“只支持 SSLIOP 协议”两个选项。SSL 可以让客户机和服务器之间建立一条加密的安全通道，能保证所传输的信息不被他人非法窃取。如果选择只支持 SSLIOP 协议，则必须进行步骤十二配置 SSL 功能。
- 使用 SSL 协议需要设置环境变量。用安装盘安装 T2000 时，会自动设置 SSL 协议的环境变量。如果不使用安装盘，需要手动建立 SSL 协议的环境变量。

PC 设置方法:

右击"我的电脑" --> 属性 --> 高级 --> 环境变量 在"系统变量"中选择 “新建” 在”变量名”中输入: *SSL_CERT_FILE* ，在"变量值"中输入: *%IMAP%\certificate\cacert.cer* -->"确定".

工作站设置方法:

确认当前用户为 t2000 用户，然后输入如下命令:

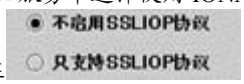
Solaris 10 下:

T2000%SSL_CERT_FILE=$IMAP/certificate/cacertcer;export SSL_CERT_FILE

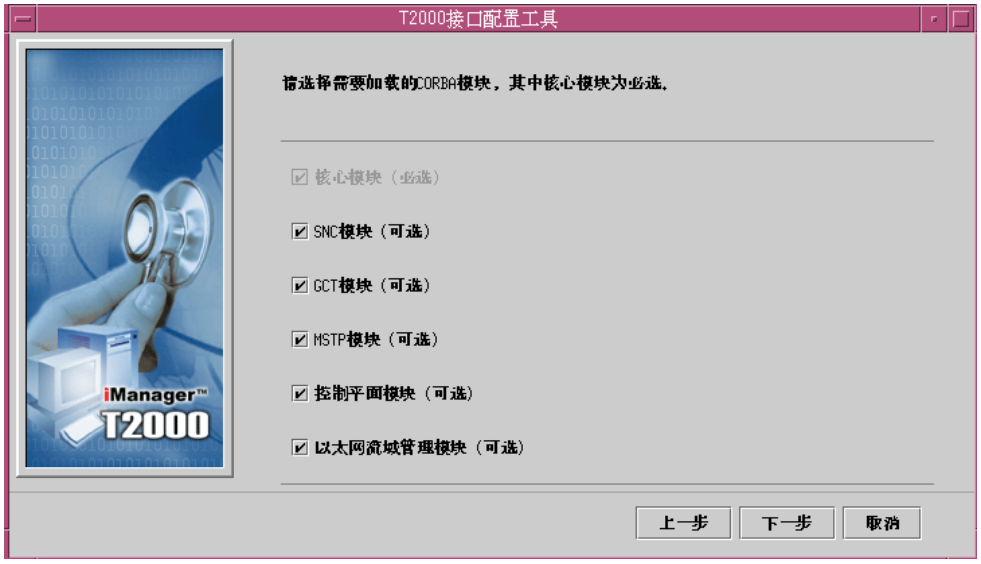
Solaris 8 下:

$SSL_CERT_FILE=$IMAP/certificate/cacertcer;export SSL_CERT_FILE

可以通过 echo $SSL_CERT_FILE 命令来查看环境变量是否设置正确。

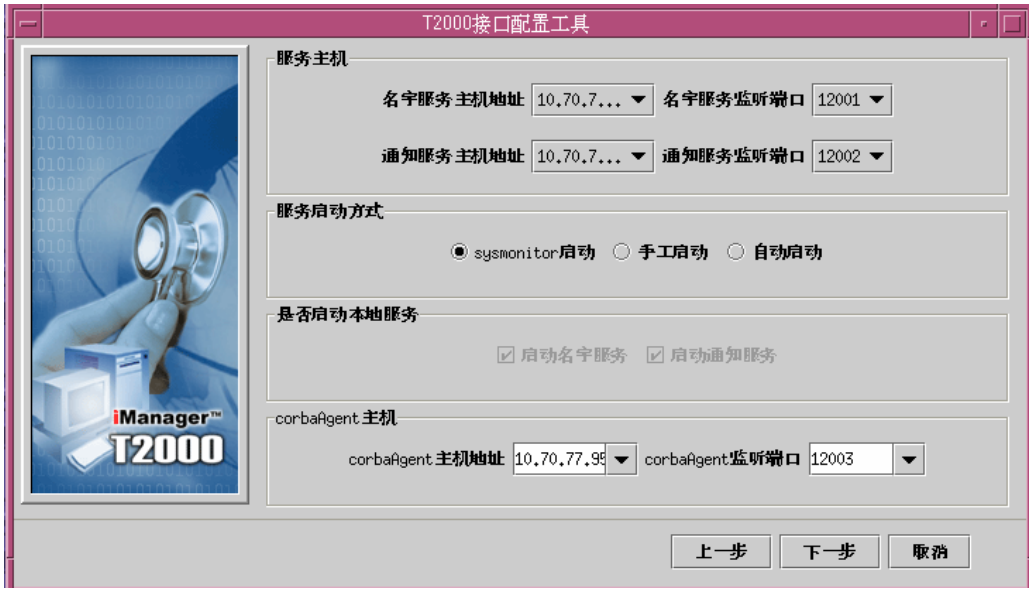
- 如果在 5.1.3 *检查通知服务*中选择使用 IONA 的 orbix 通知服务，由于 orbix 不支持 ssl 协议，所以在界面上不存在 

选项。如下图



步骤 4 配置 CORBA 服务。

1. 如果在 *5.1.3 检查通知服务*中选择使用 TAO 通知服务，则配置工具界面如下图：



在"服务主机"窗格中设置参数，名字服务主机地址和通知服务主机地址支持 IP 和主机名方式配置。

说明

- 名字服务主机地址和通知服务主机地址都不能使用"localhost"。
- 名字服务监听端口和通知服务监听端口采用默认值即可。
- 如果在"名字服务主机地址"或"通知服务主机地址"中输入了"localhost"，单击<下一步>，将弹出对话框，提示用户选择实际 IP 地址和主机名进行替换。

在"服务启动方式"窗格中设置参数，选择 sysmonitor 启动。

在”是否启动本地服务”采用默认配置即可。

2. 如果在 *5.1.3 检查通知服务*中选择使用 IONA 的 orbix 通知服务，则配置工具界面如下图：

在"服务主机"窗格中设置参数，名字服务主机地址和通知服务主机地址支持 IP 配置方式。

说明

- 名字服务主机地址和通知服务主机地址都不能使用"localhost"，可以在下拉框中选择本机的 IP。
- 名字服务监听端口和通知服务监听端口采用默认值即可。
- 如果在"名字服务主机地址"或"通知服务主机地址"中输入了"localhost"，单击<下一步>，将弹出对话框，提示用户选择实际 IP 地址进行替换。

在”服务启动方式”窗格中设置参数，选择 sysmonitor 启动。

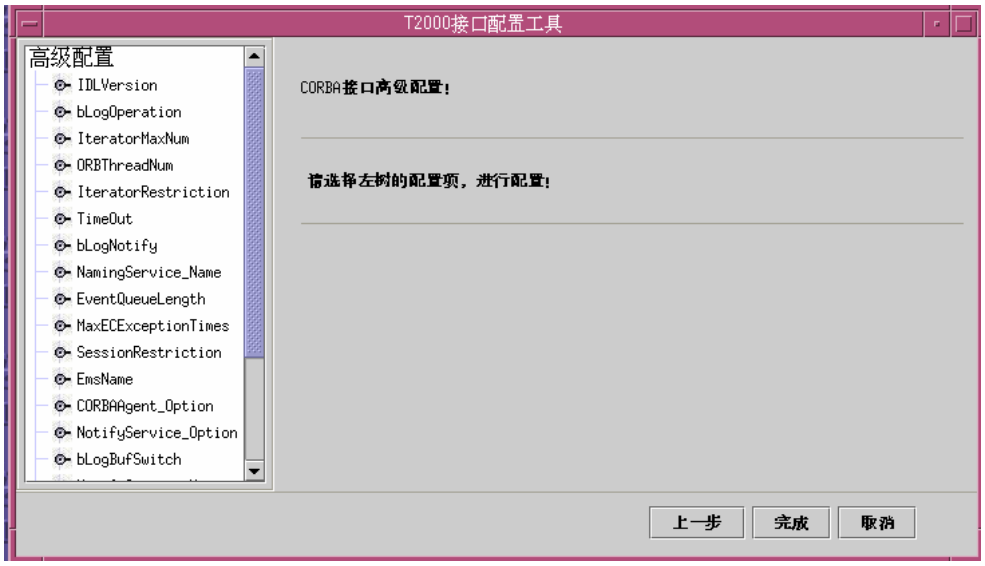
在”立即启动本地服务”采用默认配置即可。

步骤 5 在"corbaAgent 主机"窗格中设置参数，单击<下一步>。

说明

- corbaAgent 主机地址不能使用"localhost"或”127.0.0.1”，可以使用主机名称或主机 IP。
- corbaAgent 监听端口采用默认值即可。
- 如果在"corbaAgent 主机地址"中输入了"localhost"或”127.0.0.1”，单击<下一步>，将弹出对话框，提示用户选择实际 IP 地址或主机名替换。

步骤 6 进行接口的高级配置。



选择左边的配置项，可以进行高级配置(也可以直接采用默认值，而不进行配置)，这里进行配置：

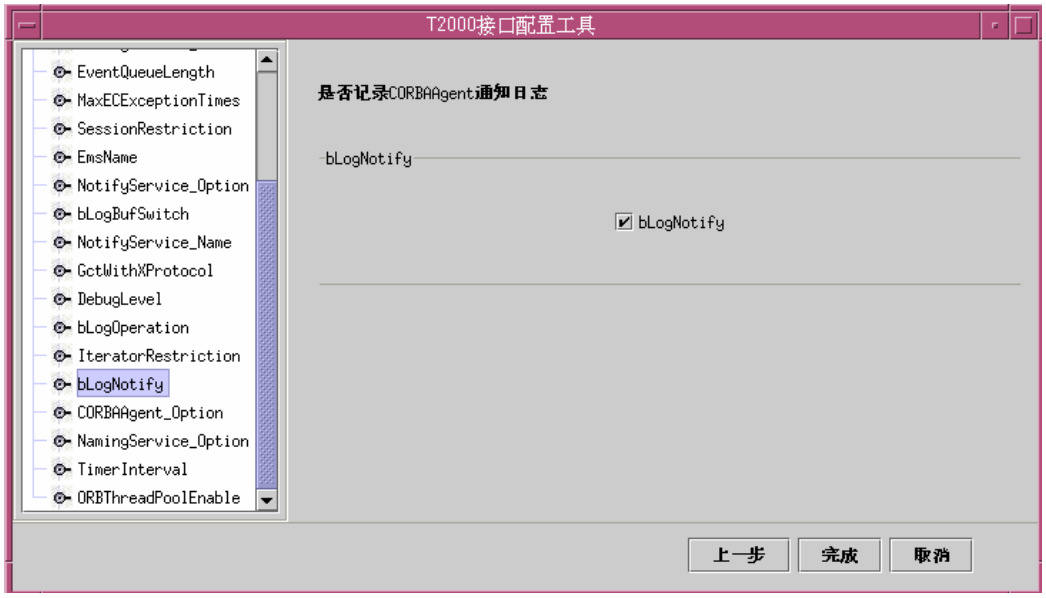
- bLogOperation：操作日志开关
- bLogNotify：通知日志开关
- bLogBufSwitch：日志缓冲
- EmsName：全网唯一标识
- Debug Level：日志级别

步骤 7 选择"bLogOperation"，出现如下界面。

说明

选择是否将所有操作写入日志。选中代表接口操作将记录在日志文件中，不选中表示不记录，默认为选中。

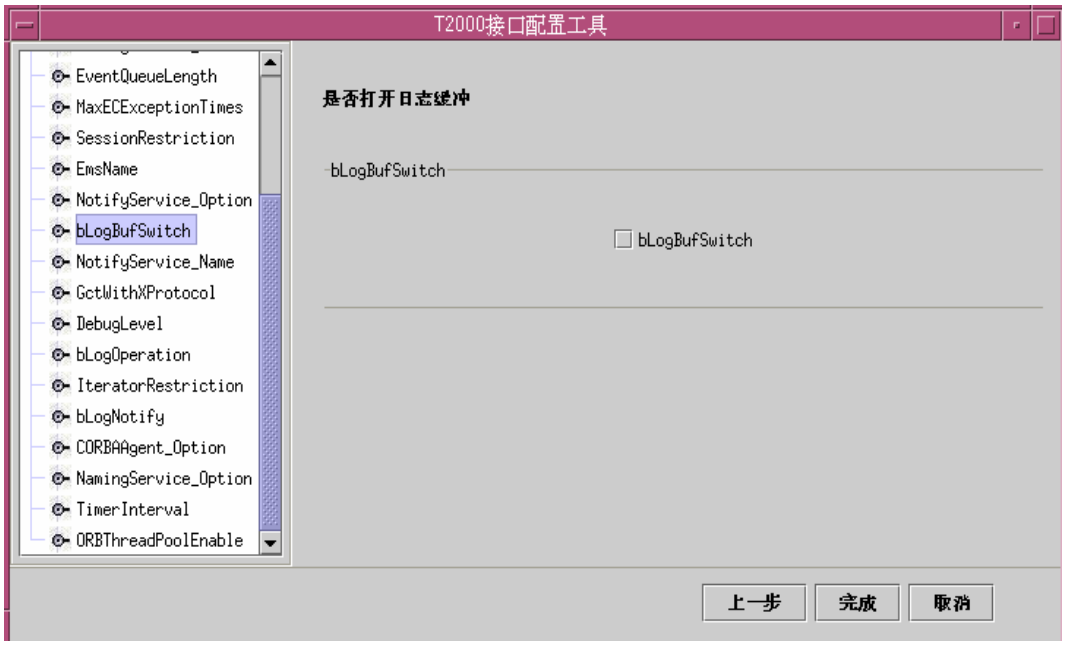
步骤 8 选择"bLogNotify"，出现如下界面。



说明

选择是否将上报通知写入日志。选中代表接口通知事件将记录在日志文件中，不选中表示不记录，默认为选中。

步骤 9 选择"bLogBufSwitch"，出现如下界面。



说明

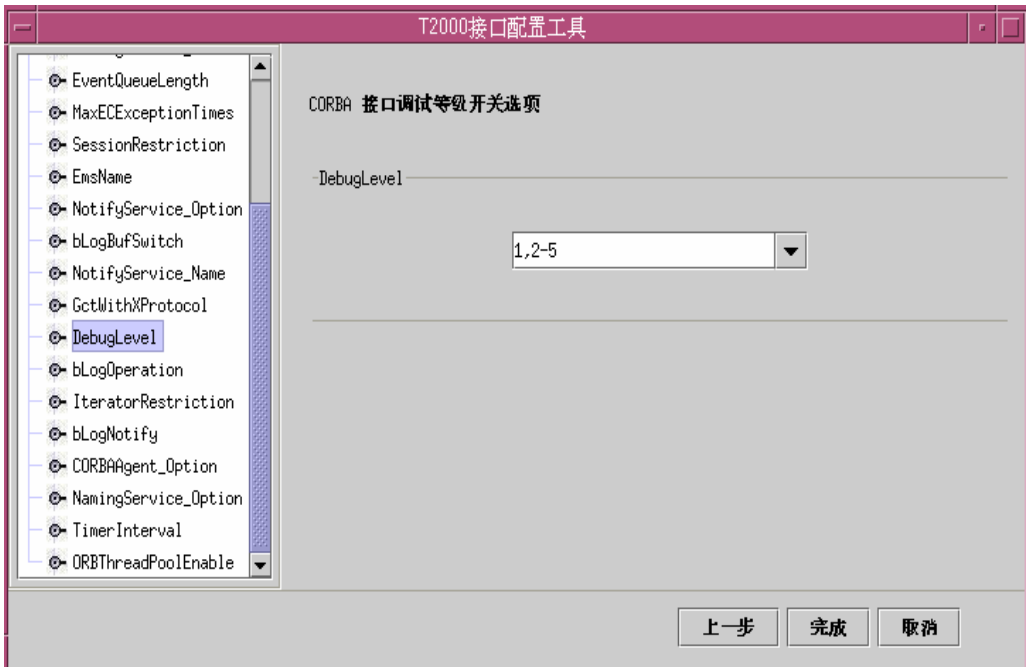
选择是否实时写入日志。选中代表事件不会实时写入到日志文件中，不选中表示实时写入，默认为不选中。

步骤 10 选择"EmsName"，出现如下界面。

说明

当网络规划中，需要配置多个 T2000 系统时，需要修改此配置参数，以确保不同的 T2000 系统的名字在 NMS 管理域内是唯一的。默认值是"Huawei/T2000"。

步骤 11 选择"Debug Level"，出现如下界面。



根据错误类型的不同，日志可以划分为 5 种级别：($IMIAP/corba.cfg 中有相应描述)

- 1st switch：CORBA platform error
- 2nd switch：Assert/Internal error
- 3rd switch：External error
- 4th switch：Function input/output
- 5th switch：Debug trace

可以根据需要，选择写入日志的错误类型。

日志级别的输入格式有比较严格的要求：

- 如果只选择单个级别且不为 1，则所选级别前所有位要用 0 补齐，用逗号分隔。例如：只选择级别 3，那么需要输入 0,0,3。
- 如果选择连续级别，可用横线连接。若不是从级别 1 开始，前面所有位仍要用 0 补齐，以逗号分隔。例如：选择级别 2,3,4，则可输入 0,2-4；选择级别 1,2,3,4,5，可以输入 1,2,3,4,5 或者 1-5。
- 如果选择不连续级别，空余位仍要用 0 补齐，以逗号分隔。例如：选择级别 1,3,5，则输入：1,0,3,0,5；选择级别 2,4，则输入：0,2,0,4。

步骤 12 单击<下一步>，配置 SSL 参数，直接使用默认值即可。

说明

如果在步骤四选中"不启用 SSLIOP 协议"，则不会出现此步，直接进入步骤十三。

步骤 13 单击<完成>，结束配置。

至此 CORBA 接口软件的配置就完成了。

----结束

启动 T2000 网管系统

Solaris 下使用 t2000 用户登录网管系统，Windows 下使用 Administrator 用户登录网管系统。

Solaris 下启动$IMAP/bin 下的 t2000server。

命令：

```
sh t2000server
```

输入用户名和密码，最后登录。如下图：

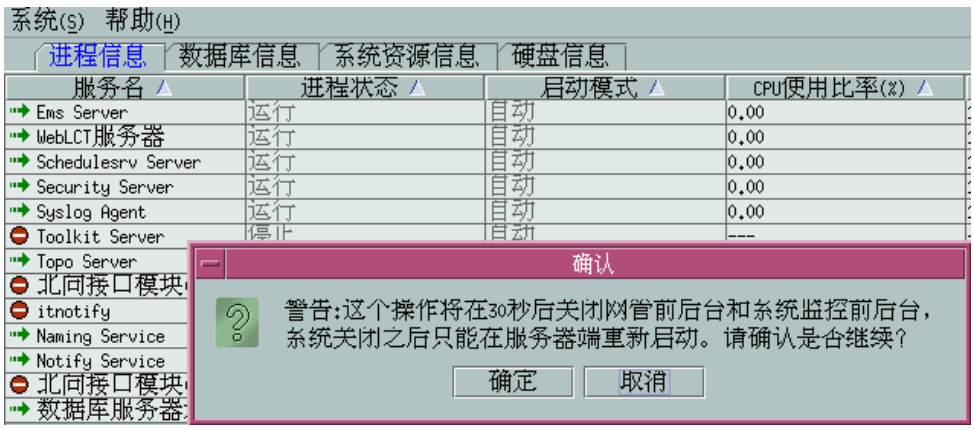
至此 T2000server(加载 CORBA 接口软件)就已经启动完成了(Windows 下直接双击 %IMAP%\bin\t2000server.bat 即可启动 T2000 网管系统)。

5.2.2 重新启动 T2000 CORBA 接口

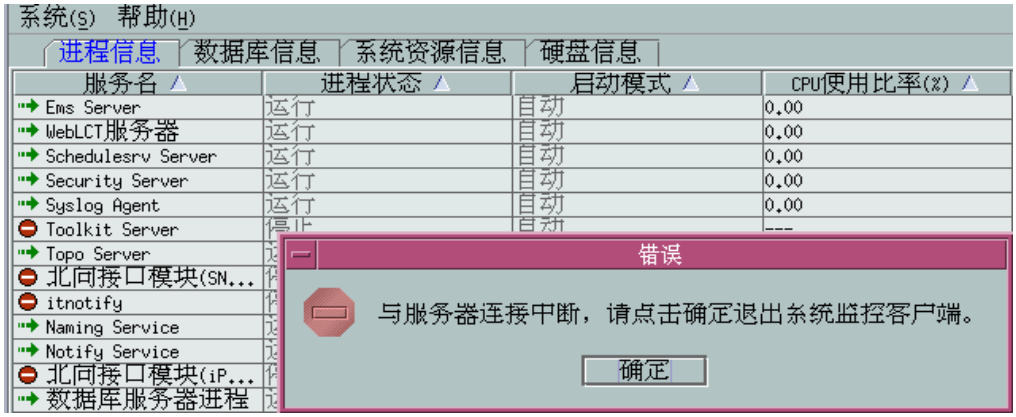
建议按照如下步骤重新启动 CORBA 接口。

步骤 1 登录 T2000 系统监控客户端，关闭网管系统，然后退出 T2000 系统监控客户端，如下图：

接下来弹出确认对话框如下图：



30 秒后 T2000 Server 系统最终关闭，点击确认即可退出监控系统客户端



步骤 2 如果重新配置了 IP 地址或者需要重新进行 T2000 CORBA 接口配置，则需要参照上述使用 T2000 配置工具对 CORBA 接口进行配置的步骤进行重新配置(可选)。

说明

如果重新配置了 IP 地址则还必须要删除目录$IMAP 下原来生成的持久化文件：ns.ior 和 persistence.ns，否则 T2000 Server 将无法正常启动。

步骤 3 启动$IMAP/bin 下的 t2000server (具体操作请参考 5.2.1 启动 T2000 网管系统部分)。

----结束

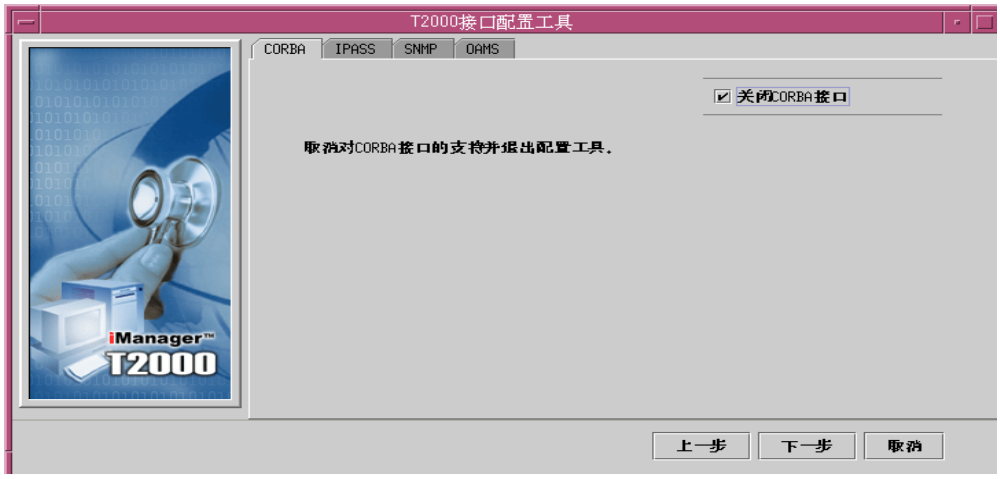
5.2.3 禁用 T2000 CORBA 接口

建议按照如下步骤禁用 T2000 CORBA 接口。

如果在 T2000 系统运行过程中不需要再提供 CORBA 接口功能，可以按照下面步骤来关闭 T2000 CORBA 接口功能。

步骤 1 登录 T2000 系统监控进程客户端，关闭 T2000 网管系统，退出 T2000 系统监控客户端(具体操作可参见 5.2.2 重新启动 T2000 CORBA 接口第一步中的操作)。

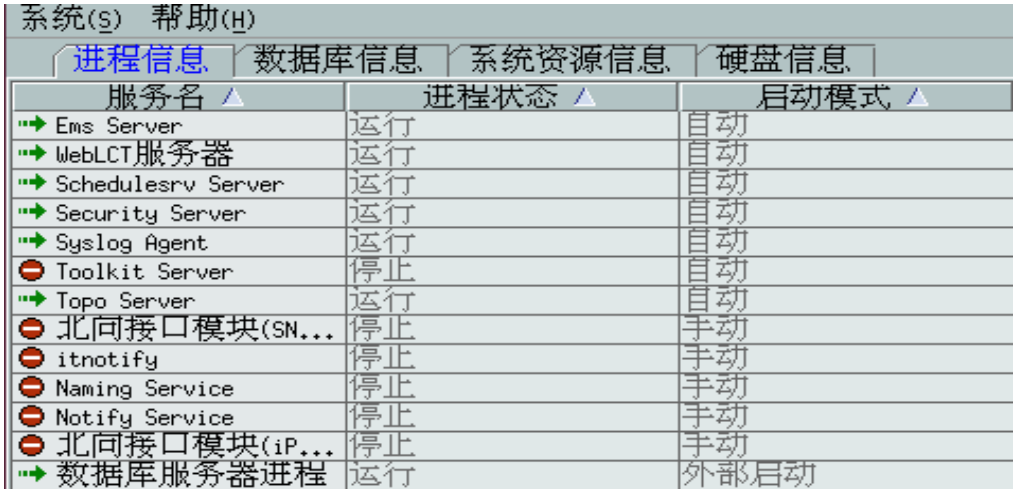
步骤 2 通过 T2000 配置工具关闭 CORBA 接口，如下图：



说明

如何启动 T2000 配置工具请参看 5.2.1.1 使用 T2000 配置工具对 CORBA 接口进行配置部分。

步骤 3 启动$IMAP/bin 下的 t2000server，进而启动 T2000 网管系统。系统启动后如下图所示：



说明

此时 Naming Service 和 Notify Service 已经关闭，此时 CORBA 接口已经被禁用。

----结束

5.3 启动和禁用 CORBA 接口(双机)

Solaris 下支持双机环境，Windows 下不支持双机环境。本文中关于启动和禁用 CORBA 接口的描述分为 Solaris10 系统 和 Solaris 8 系统两部分进行。

注意

本文中提到的用户切换命令为:

`su - user`(user 为需要切换的用户名)

然后根据提示输入密码即可进行用户的切换，下文涉及到用户切换部分不再进行描述。

5.3.1 Solaris 10 系统

请使用 t2000 用户登录 Solaris 10 系统。

启动 T2000 CORBA 接口

分别在主机和备机上启动 T2000 CORBA 接口，过程中需进行如下操作：

- 双机配置
- 使用 T2000 配置工具对 CORBA 接口进行配置
- 启动 T2000 网管系统

说明

在主机和备机上使用 T2000 配置工具对 CORBA 接口进行配置的步骤完全一致，这里举在主机上进行配置为例。

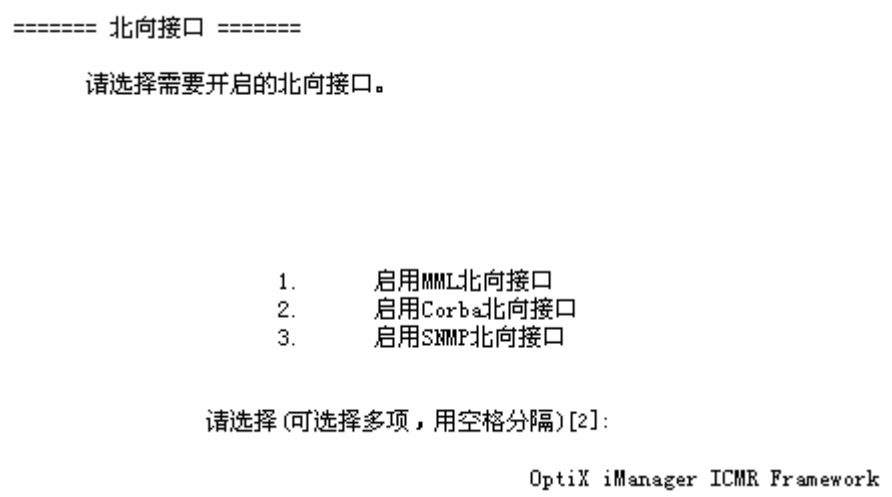
双机配置

步骤 1 打开一个终端控制台窗口，需要修改窗口大小。选择：选项 --> 窗口大小 --> 132×24，如果当前用户不是 root 用户，请切换到 root 用户。

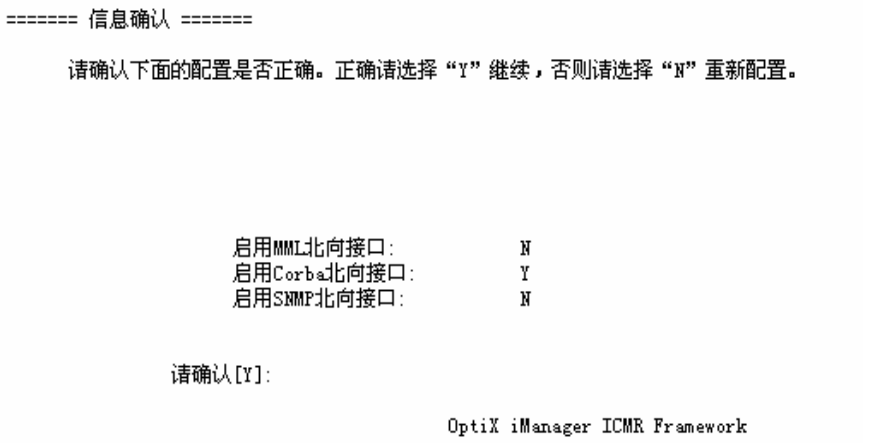
步骤 2 在主用服务器和备用服务器运行激活北向接口命令。

```
#cd /opt/HWICMR/bin
#./runtaskflow.sh config_nbi_ha.tf
```

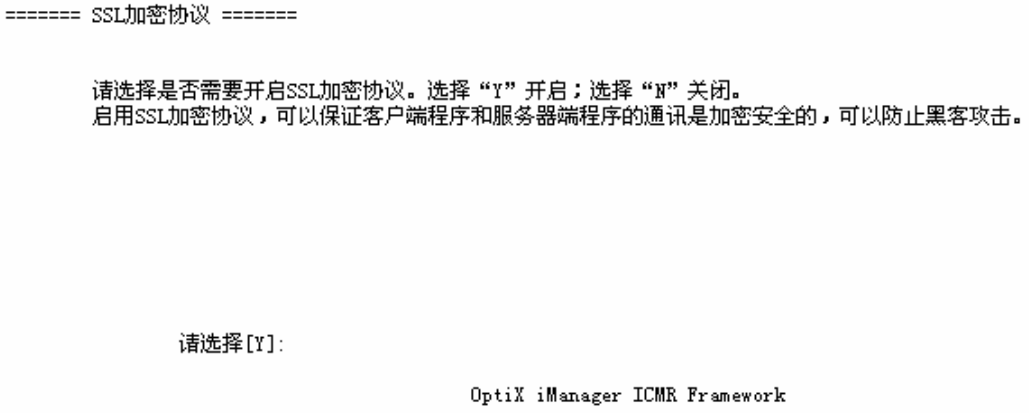
步骤 3 选择需要启用的北向接口。如下图所示，输入 2，选择启用 CORBA 北向接口。



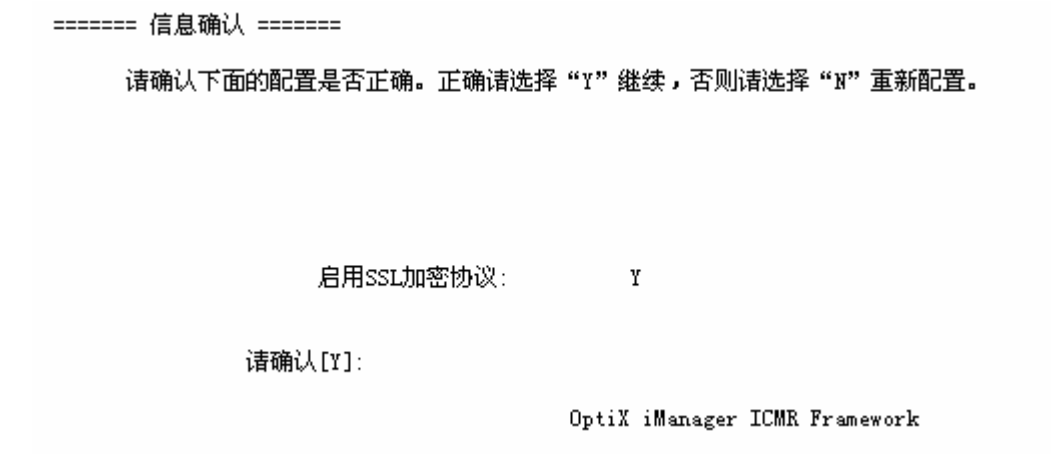
步骤 4 选择确认，如下图所示，输入 N 或 Y 表示重新配置和确认。



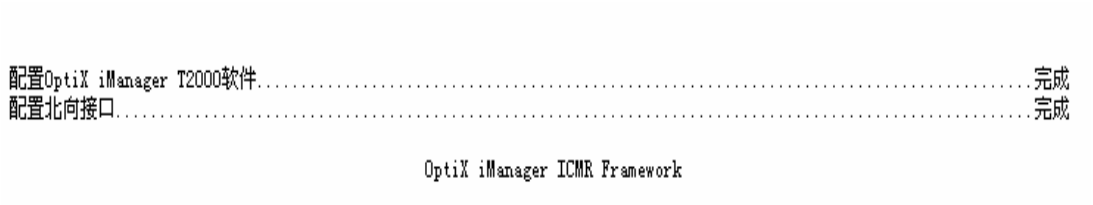
步骤 5 选择是否启用 SSL 加密协议。如下图所示，输入 N 或 Y 表示不启用和启用。



步骤 6 配置信息确认，如下图所示，输入 N 或 Y 表示不确认需重新配置和确认已配置。



步骤 7 完成配置，见下图。



----结束

使用 T2000 配置工具对 CORBA 接口进行配置

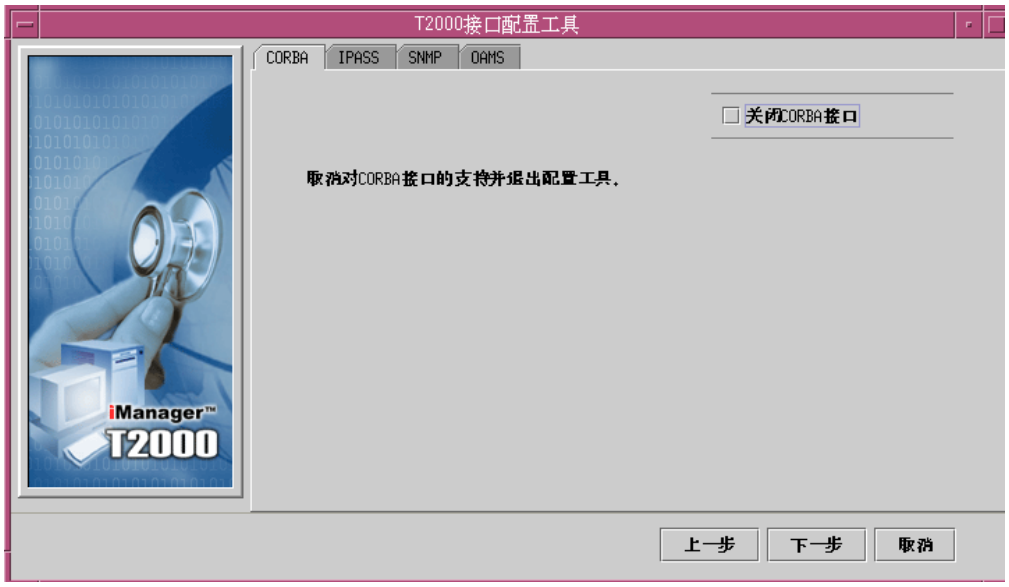
配置工具可以自动识别 T2000 的版本、语言。不需要做任何配置，直接运行$IMAP/tools/configtool/configtool.sh 即可使用配置工具。建议按照以下步骤使用配置工具配置 CORBA。

步骤 1 如果此时用户不是 t2000 用户，请切换到 t2000 用户，启动配置工具 $IMIAP/tools/configtool/configtool.sh。

Solaris 下执行如下命令来启动工具。

```
sh configtool.sh
```

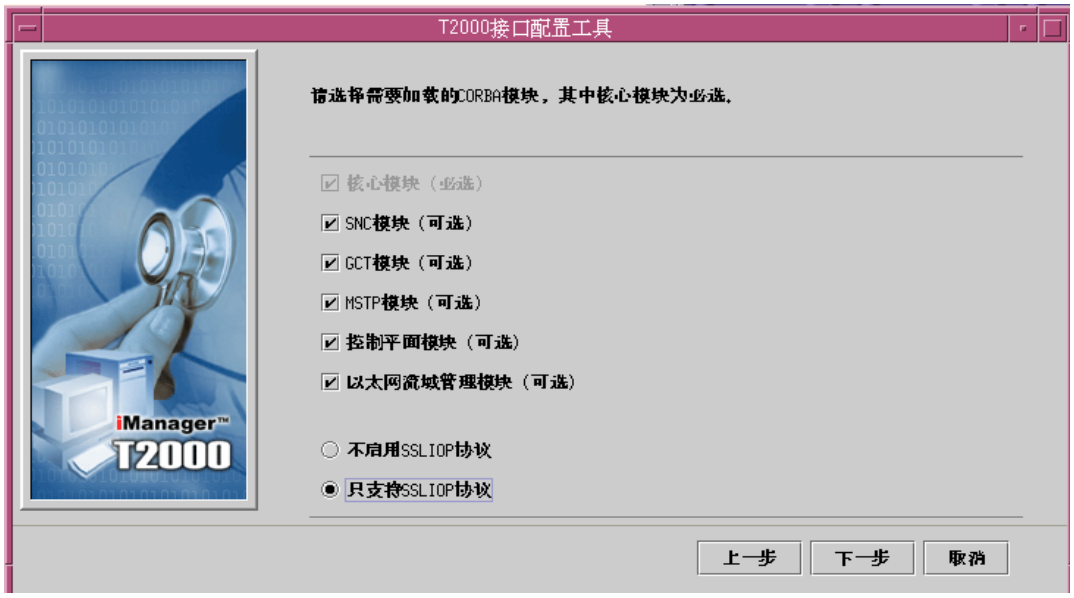
工具启动后，单击<下一步>。



步骤 2 单击<下一步>。

说明

如果选中"关闭 CORBA 接口"功能，单击<下一步>将退出 T2000 配置工具。



步骤 3 配置需要加载的 CORBA 模块，单击<下一步>。

注意

如果在双机配置中选择启用 SSL 协议，则在这里需要选择”只支持 SSLIOP 协议”项。如果在双机配置中选择不启用 SSL 协议，则在这里需要选择”不启用 SSLIOP 协议”项。

说明

- 这里可以选择要加载 CORBA 的模块。其中"核心模块"为必选模块，其余模块可以根据已有 license 支持情况选择加载。
- 对于 T2000V2R3C01 及以后版本，会出现“不启用 SSLIOP 协议”和“只支持 SSLIOP 协议”两个选项。SSL 可以让客户机和服务器之间建立一条加密的安全通道，能保证所传输的信息不被他人非法窃取。如果选择只支持 SSLIOP 协议，则必须进行步骤十二配置 SSL 功能。
- 使用 SSL 协议需要设置环境变量。用安装盘安装 T2000 时，会自动设置 SSL 协议的环境变量。如果不使用安装盘，需要手动设置 SSL 协议的环境变量。

工作站设置方法:

确认当前用户为 t2000 用户，然后输入如下命令:

T2000%*SSL_CERT_FILE=$IMAP/certificate/cacertcer;export SSL_CERT_FILE*

可以通过 echo $ SSL_CERT_FILE 命令来查看环境变量是否设置正确。

步骤 4 配置 CORBA 的服务。

说明

其中主机地址 10.70.77.95 为示例地址，需根据实际主机地址作调整。

在"服务主机"窗格中设置参数，名字服务主机地址和通知服务主机地址支持 IP 和主机名方式配置。

说明

- 名字服务主机地址和通知服务主机地址都不能使用"localhost"。
- 名字服务监听端口和通知服务监听端口采用默认值即可。
- 如果在"名字服务主机地址"或"通知服务主机地址"中输入了"localhost"，单击<下一步>，将弹出对话框，提示用户选择实际 IP 地址和主机名进行替换。

在"服务启动方式"窗格中设置参数，选择 sysmonitor 启动。

在”是否启动本地服务”采用默认配置即可。

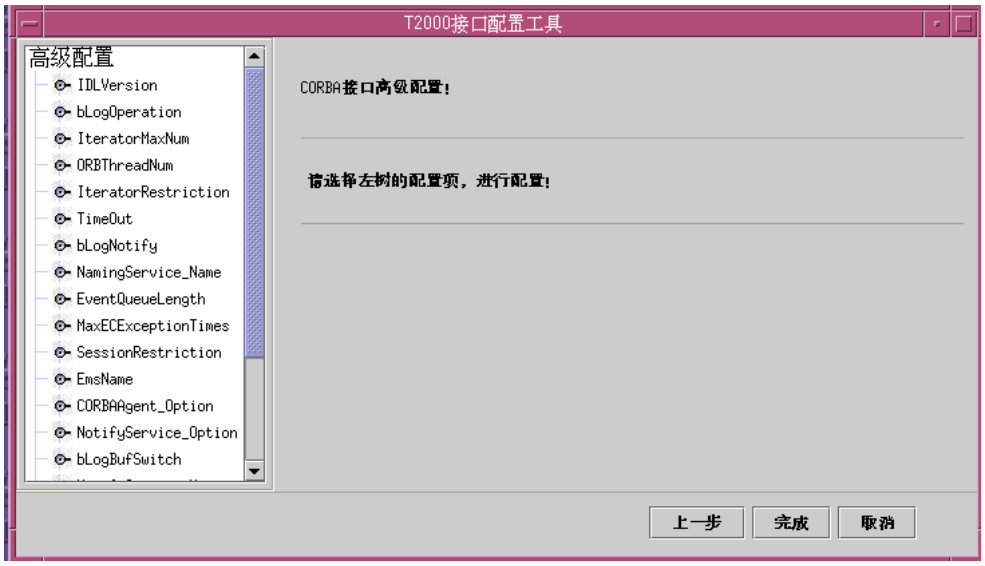
步骤 5 在"corbaAgent 主机"窗格中设置参数。

说明

corbaAgent 主机地址不能使用"localhost"或"127.0.0.1"，可以使用主机名称或主机 IP。

如果在"corbaAgent 主机地址"中输入了"localhost"或"127.0.0.1"， 单击<下一步>，将弹出对话框，提示用户选择实际 IP 地址替换。

步骤 6 单击<下一步>，进行接口的高级配置。



选择左边的配置项，可以进行高级配置(也可以直接采用默认值，而不进行配置)，这里进行配置：

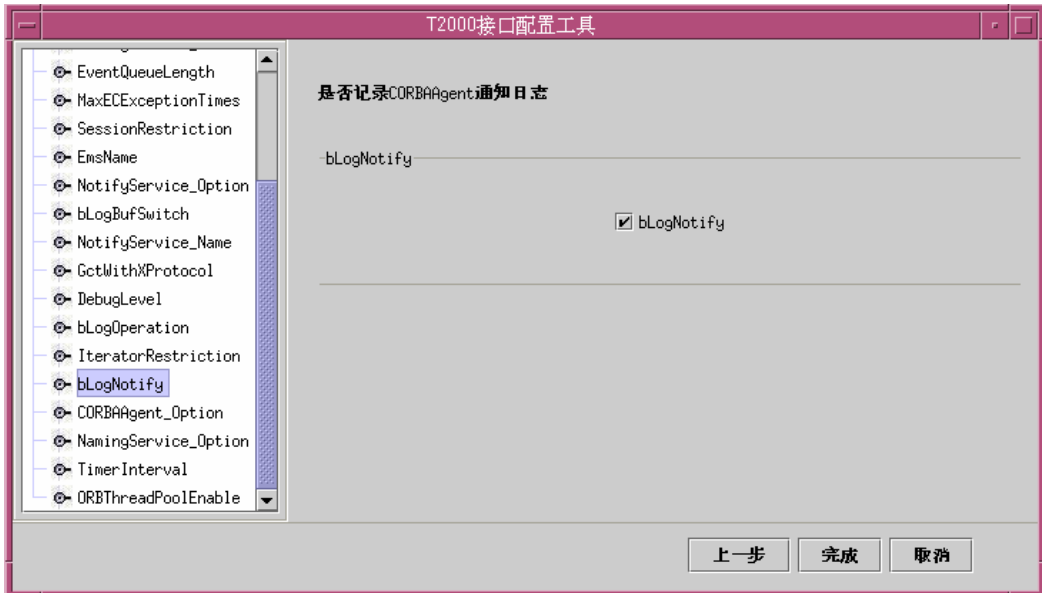
- bLogOperation：操作日志开关
- bLogNotify：通知日志开关
- bLogBufSwitch：日志缓冲
- EmsName：全网唯一标识
- Debug Level：日志级别

步骤 7 选择"bLogOperation"，出现如下界面。

说明

选择是否将所有操作写入日志。选中代表接口操作将记录在日志文件中，不选中表示不记录，默认为选中。

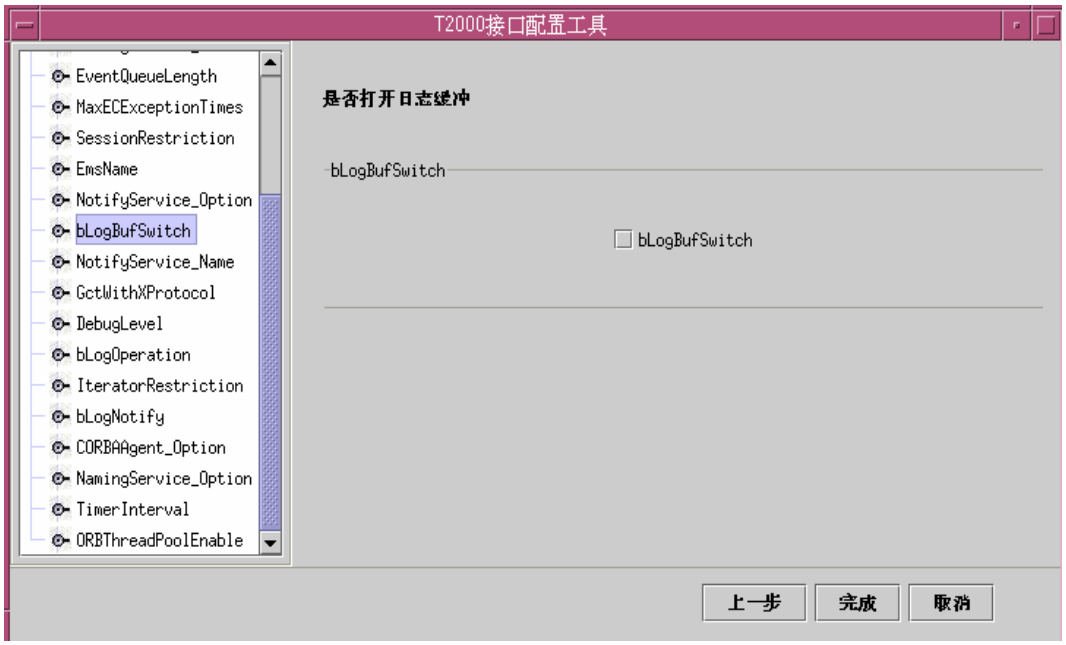
步骤 8 选择"bLogNotify"，出现如下界面。



说明

选择是否将上报通知写入日志。选中代表接口通知事件将记录在日志文件中，不选中表示不记录，默认为选中。

步骤 9 选择"bLogBufSwitch"，出现如下界面。



说明

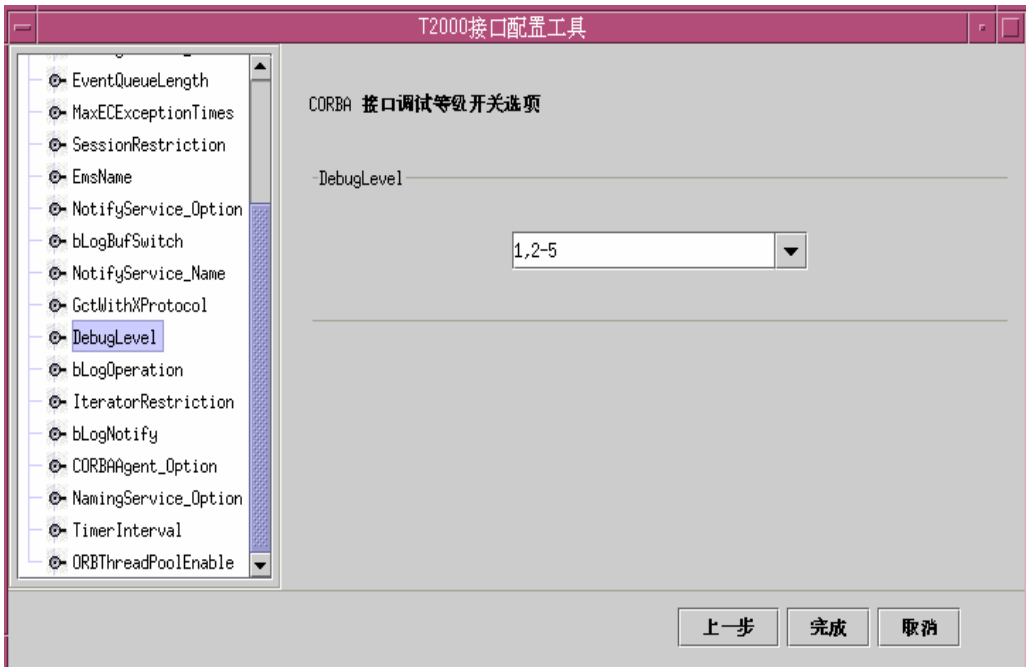
选择是否实时写入日志。选中代表事件不会实时写入到日志文件中，不选中表示实时写入，默认为不选中。

步骤 10 选择"EmsName"，出现如下界面。

说明

当网络规划中，需要配置多个 T2000 系统时，需要修改此配置参数，以确保不同的 T2000 系统的名字在 NMS 管理域内是唯一的。默认值是"Huawei/T2000"。

步骤 11 选择"Debug Level"，出现如下界面。



根据错误类型的不同，日志可以划分为 5 种级别：($IMIAP/corba.cfg 中有相应描述)

- 1st switch：CORBA platform error
- 2nd switch：Assert/Internal error
- 3rd switch：External error
- 4th switch：Function input/output
- 5th switch：Debug trace

可以根据需要，选择写入日志的错误类型。

日志级别的输入格式有比较严格的要求：

- 如果只选择单个级别且不为 1，则所选级别前所有位要用 0 补齐，用逗号分隔。例如：只选择级别 3，那么需要输入 0,0,3。
- 如果选择连续级别，可用横线连接。若不是从级别 1 开始，前面所有位仍要用 0 补齐，以逗号分隔。例如：选择级别 2,3,4，则可输入 0,2-4；选择级别 1,2,3,4,5，可以输入 1,2,3,4,5 或者 1-5。
- 如果选择不连续级别，空余位仍要用 0 补齐，以逗号分隔。例如：选择级别 1,3,5，则输入：1,0,3,0,5；选择级别 2,4，则输入：0,2,0,4。

步骤 12 单击<下一步>，配置 SSL 参数，直接使用默认值即可。

说明

如果在步骤四选中"不启用 SSLIOP 协议"，跳过此步，进入步骤十三。

步骤 13 单击<完成>，结束配置。

至此 CORBA 接口软件的配置就完成了。

----结束

启用 T2000 网管系统

如果是 Veritas 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令启动主用服务器 T2000 网管系统：

```
#hagrp -online AppService -sys 主用服务器主机名称
```

如果是 Veritas 温备份高可用性系统：

如果当前用户不是 t2000 用户，请切换到 t2000 用户并执行如下命令启动主用服务器 T2000 网管系统：

```
T2000%cd $IMAP/bin
T2000%./t2000server
```

如果是 SunCluster 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令启动主用服务器 T2000 网管系统：

```
#scswitch -Z -g AppService
```

如果是 Watchman 热备份高可用性系统：

按照如下方式启动主用服务器 T2000 网管系统：

使用 Watchman 客户端登陆主用服务器，单击客户端的<启动应用>按键。

重新启动 T2000 CORBA 接口

建议按照以下步骤重新启动 CORBA 接口。

如果是 Veritas 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下步骤：

步骤 1 关闭主用服务器 T2000 网管系统。

```
#hagrp -offline AppService -sys 主用服务器主机名称
```

步骤 2 启动主用服务器 T2000 网管系统。

```
#hagrp -online AppService -sys 主用服务器主机名称
```

----结束

如果是 Veritas 温备份高可用性系统：

如果当前用户不是 t2000 用户，请切换到 t2000 用户并执行如下步骤：

步骤 1 步骤一：关闭主用服务器 T2000 网管系统。

```
T2000%cd $IMAP/bin
T2000%./shutdownserver
```

步骤 2 启动主用服务器 T2000 网管系统。

```
T2000%cd $IMAP/bin
T2000%./t2000server
```

----结束

如果是 SunCluster 热备份高可用性系统：

如果当前不是 root 用户，请切换到 root 用户并执行如下步骤：

步骤 1 关闭主用服务器 T2000 网管系统。

```
#scswitch -F -g AppService
```

步骤 2 启动主用服务器 T2000 网管系统。

```
#scswitch -Z -g AppService
```

----结束

如果是 Watchman 热备份高可用性系统：

步骤 1 关闭主用服务器 T2000 网管系统。

使用 Watchman 客户端登陆主用服务器，单击客户端的<关闭应用>按键。

步骤 2 启动主用服务器 T2000 网管系统。

使用 Watchman 客户端登陆主用服务器，单击客户端的<启动应用>按键。

----结束

禁用 T2000 CORBA 接口

如果在 T2000 系统运行过程中不再需要提供 CORBA 接口功能，建议按照下面步骤来关闭主用服务器 T2000 CORBA 接口功能。

步骤 1 关闭主用服务器 T2000 网管系统。

Veritas 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令：

```
#hagrp -offline AppService -sys 主用服务器主机名称
```

Veritas 温备份高可用性系统：

如果当前用户不是 t2000 用户，请切换到 t2000 用户并执行如下命令：

```
T2000%cd $IMAP/bin
T2000%./shutdownserver
```

SunCluster 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令：

```
#scswitch -F -g AppService
```

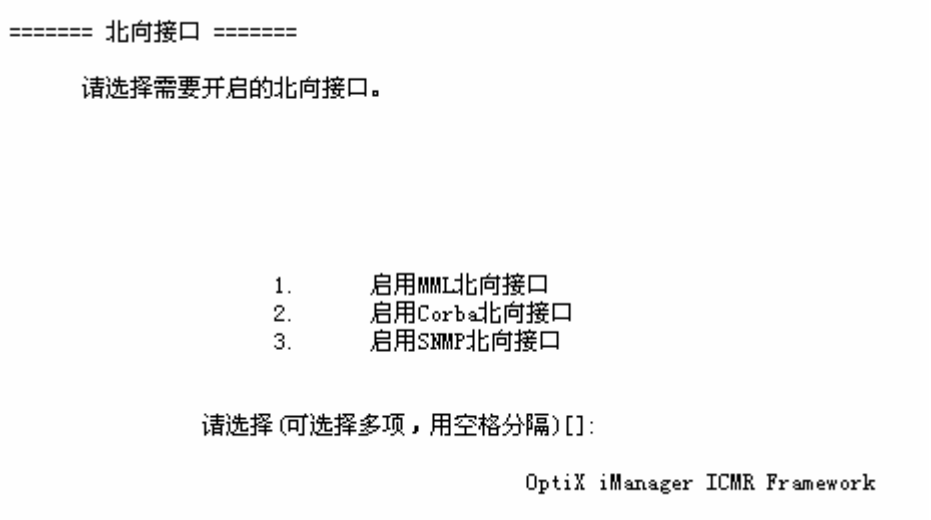
Watchman 热备份高可用性系统：

使用 Watchman 客户端登陆主用服务器，单击客户端的<关闭应用>按键。

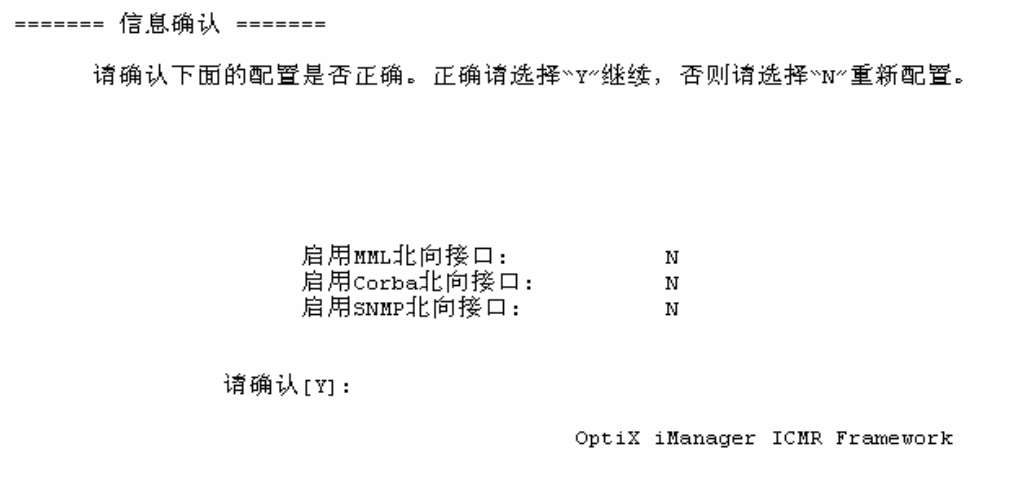
步骤 2 打开一个终端控制台窗口，需要修改窗口大小。选择：选项 --> 窗口大小 --> 132×24，如果当前用户不是 root 用户，请切换到 root 用户并在主用服务器运行北向接口命令：

```
#cd /opt/HWICMR/bin
#./runtaskflow.sh config_nbi_ha.tf
```

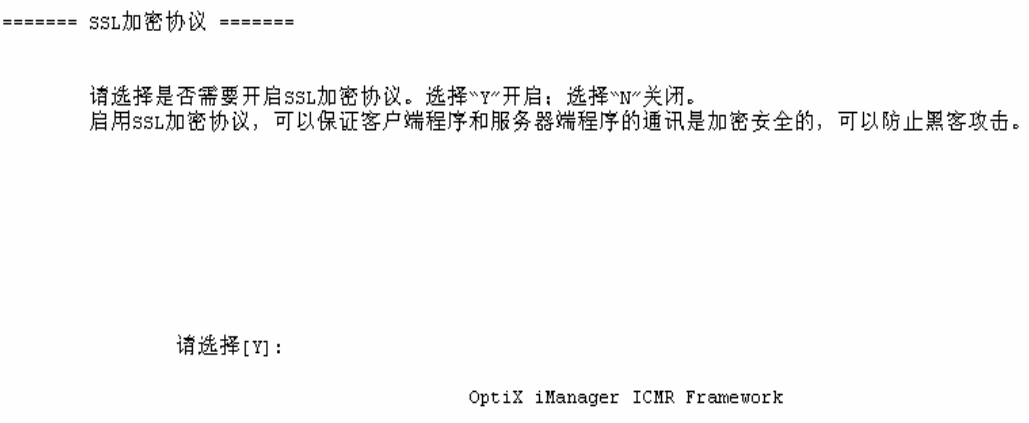
步骤 3 不选择 CORBA 接口启动，如下图所示，不输入序列号。



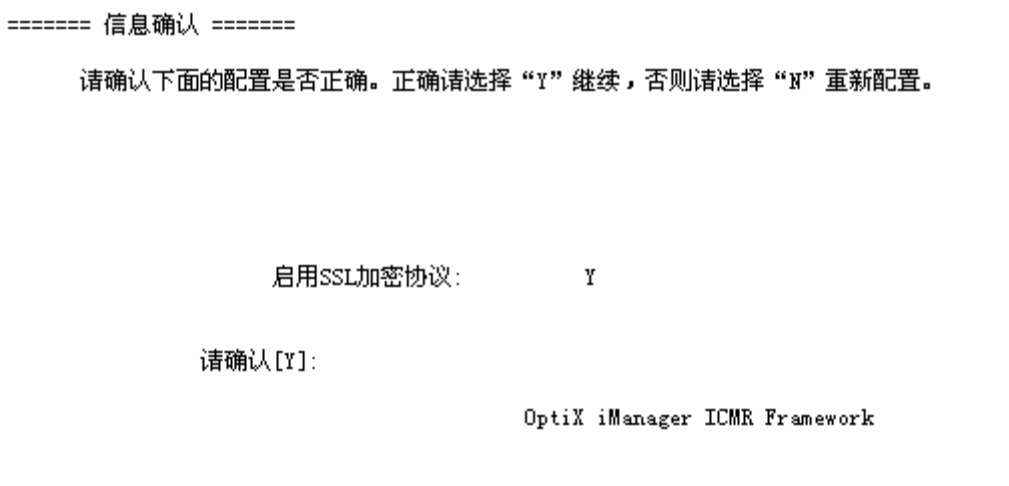
步骤 4 选择确认，如下图所示。输入 N 或 Y 表示重新配置和确认。



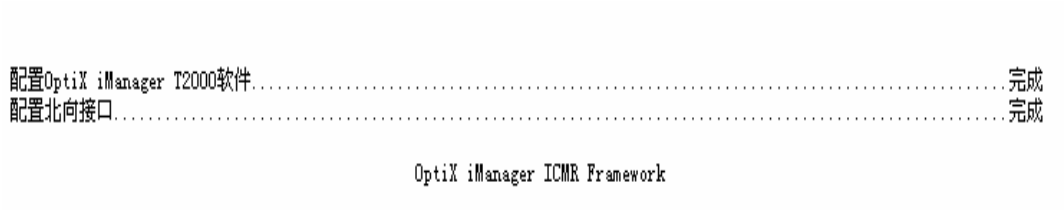
步骤 5 选择是否启用 SSL 加密协议，如下图所示。输入 N 或 Y 表示不启用和启用。



步骤 6 配置信息确认，如下图所示。输入 N 或 Y 表示不确认需重新配置和确认已配置。



步骤 7 完成配置，见下图。



步骤 8 在主用服务器启动 T2000 网管系统。

如果是 Veritas 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令：

```
#hagrp -online AppService -sys 主用服务器主机名称
```

如果是 Veritas 温备份高可用性系统：

如果当前用户不是 t2000 用户，请切换到 t2000 用户并执行如下命令：

```
T2000%cd $IMAP/bin
T2000%./t2000server
```

如果是 SunCluster 热备份高可用性系统:

如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令：

```
#scswitch -Z -g AppService
```

如果是 Watchman 热备份高可用性系统：

按照如下方式启动主用服务器 T2000 网管系统：

使用 Watchman 客户端登陆主用服务器，单击客户端的<启动应用>按键。

说明

备机禁止 T2000 CORBA 接口过程与上述过程一致，这里操作过程略。

----结束

5.3.2 Solaris 8 系统

需要使用 t2000 用户登录 Solaris 8 系统。

启动 T2000 CORBA 接口

分别在主机和备机上启动 T2000 CORBA 接口，过程中需进行如下操作：

- 使用 T2000 配置工具对 CORBA 接口进行配置
- 更新 t2000_emsstart 脚本
- 启动 T2000 网管系统

说明

在主机和备机上使用 T2000 配置工具对 CORBA 接口进行配置的步骤完全一致，这里举在主机上进行配置为例。

使用 T2000 配置工具对 CORBA 接口进行配置

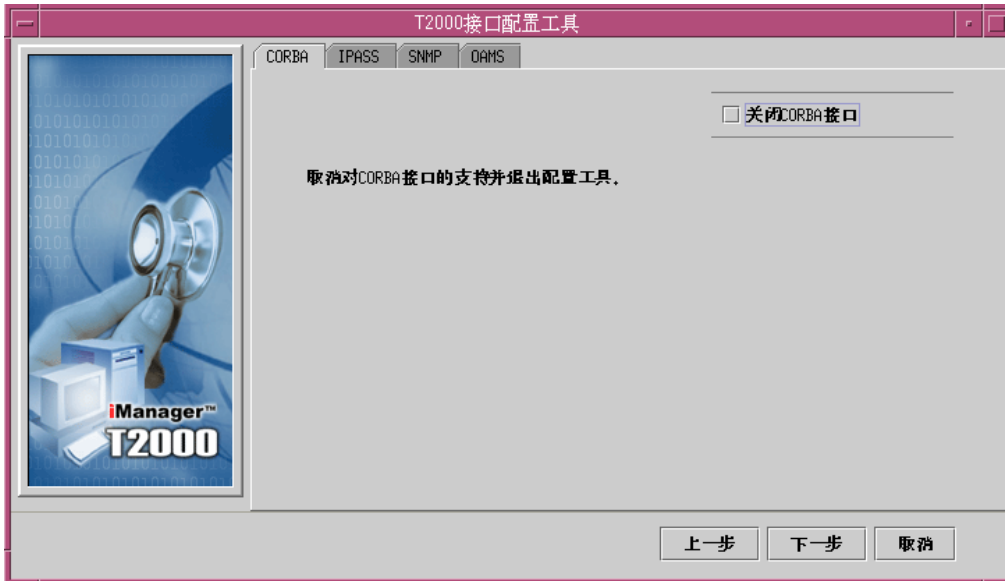
配置工具可以自动识别 T2000 的版本、语言。不需要做任何配置，直接运行$IMAP/tools/configtool/configtool.sh 即可使用配置工具。建议按照以下步骤使用配置工具配置 CORBA。

步骤 1 启动配置工具，Solaris 下使用 t2000 用户启动工具，

Solaris 下执行如下命令来启动工具。

```
sh configtool.sh
```

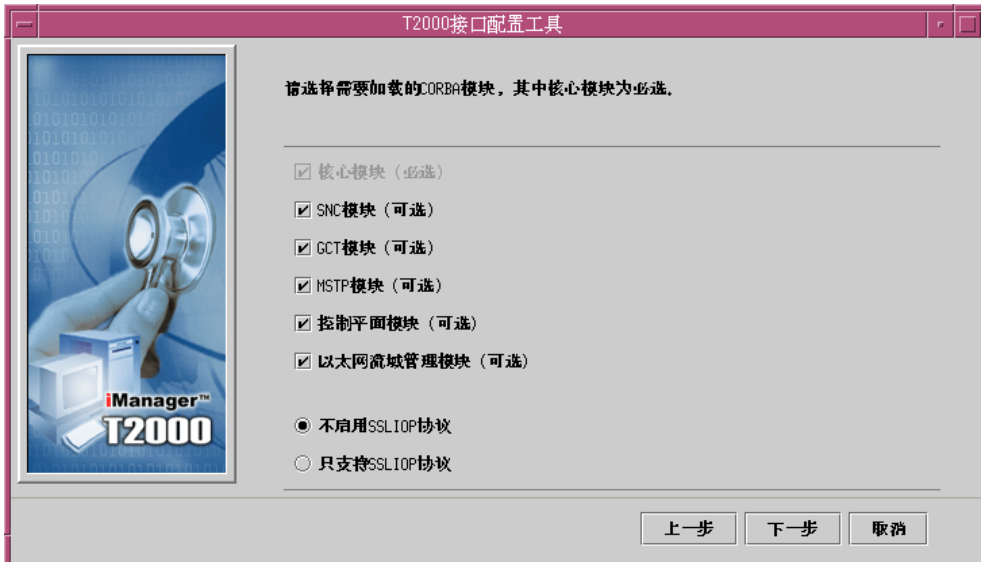
工具启动后，单击<下一步>。



步骤 2 步骤二：单击<下一步>。

说明

如果选中"关闭 CORBA 接口"功能，单击<下一步>将退出 T2000 配置工具。



步骤 3 配置需要加载的 CORBA 模块，单击<下一步>。

说明

- 这里可以选择要加载 CORBA 的模块。其中"核心模块"为必选模块，其余模块可以根据已有 license 支持情况选择加载。
- 对于 T2000V2R3C01 及以后版本，会出现“不启用 SSLIOP 协议”和“只支持 SSLIOP 协议”两个选项。SSL 可以让客户机和服务器之间建立一条加密的安全通道，能保证所传输的信息不被他人非法窃取。如果选择只支持 SSLIOP 协议，则必须进行步骤十二配置 SSL 功能。
- 使用 SSL 协议需要设置环境变量。用安装盘安装 T2000 时，会自动设置 SSL 协议的环境变量。如果不使用安装盘，需要手动设置 SSL 协议的环境变量。

工作站设置方法:

确认当前用户为 t2000 用户，然后输入如下命令:

$SSL_CERT_FILE=$IMAP/certificate/cacertcer;export SSL_CERT_FILE

可以使用 echo $ *SSL_CERT_FILE* 命令来查看环境变量是否设置正确。

步骤 4 配置 CORBA 的服务。

说明

其中主机地址 10.70.77.95 为示例地址，需根据实际主机地址作调整。

在"服务主机"窗格中设置参数，名字服务主机地址和通知服务主机地址支持 IP 和主机名方式配置。

说明

- 名字服务主机地址和通知服务主机地址都不能使用"localhost"。
- 名字服务监听端口和通知服务监听端口采用默认值即可。
- 如果在"名字服务主机地址"或"通知服务主机地址"中输入了"localhost"或”127.0.0.1”，单击<下一步>，将弹出对话框，提示用户选择实际 IP 地址和主机名进行替换。

在"服务启动方式"窗格中设置参数，选择自动启动。

在”是否启动本地服务”采用默认配置即可。

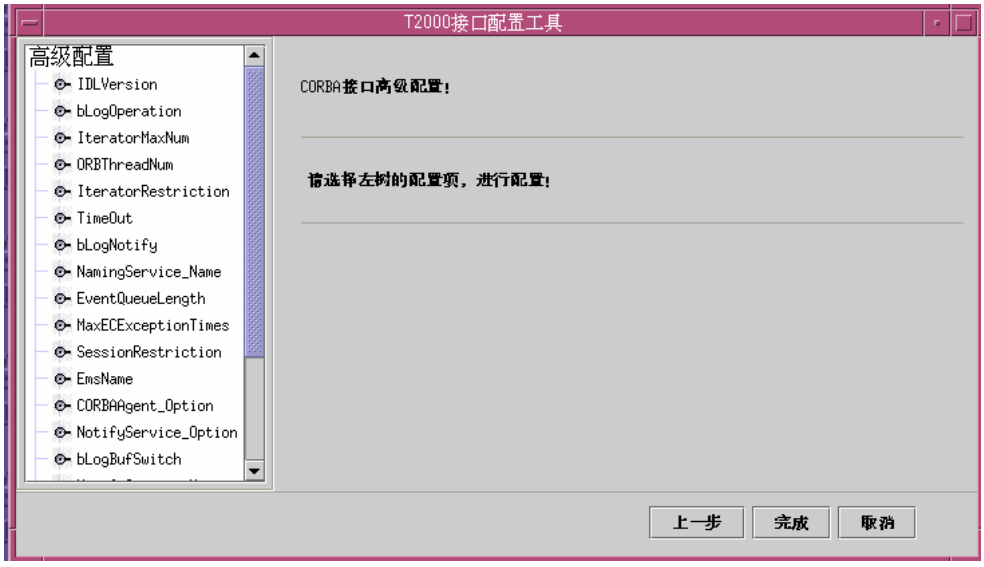
步骤 5 在"corbaAgent 主机"窗格中设置参数。

说明

corbaAgent 主机地址不能使用"localhost"或"127.0.0.1"，可以使用主机名称或主机 IP。

如果在"corbaAgent 主机地址"中输入了"localhost"或"127.0.0.1"， 单击<下一步>，将弹出对话框，提示用户选择实际 IP 地址替换。

步骤 6 单击<下一步>，进行接口的高级配置。



选择左边的配置项，可以进行高级配置(也可以直接采用默认值，而不进行配置)，这里进行配置：

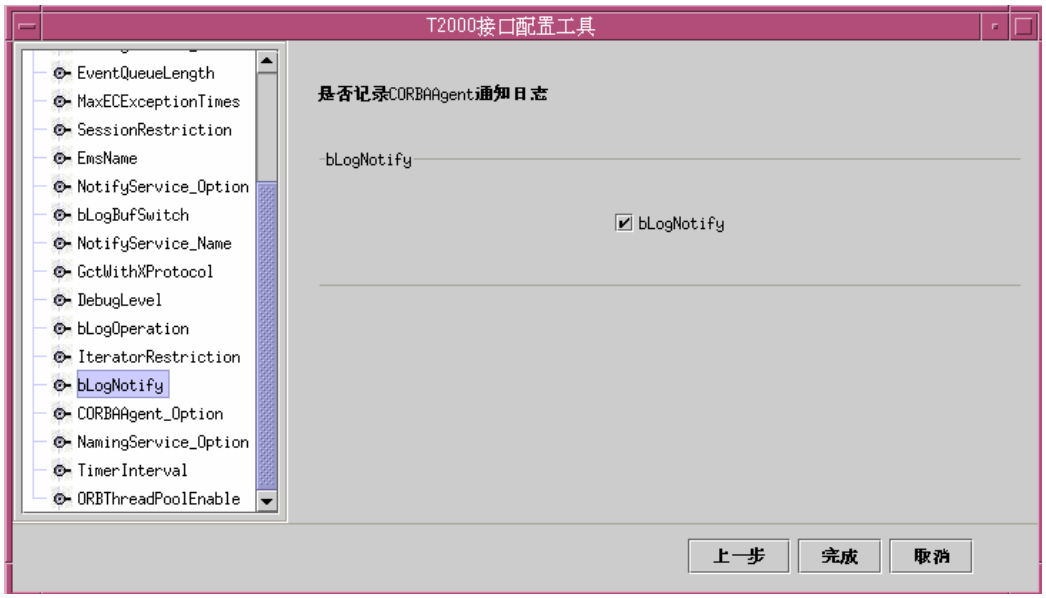
- bLogOperation：操作日志开关
- bLogNotify：通知日志开关
- bLogBufSwitch：日志缓冲
- EmsName：全网唯一标识
- Debug Level：日志级别

步骤 7 选择"bLogOperation"，出现如下界面。

说明

选择是否将所有操作写入日志。选中代表接口操作将记录在日志文件中，不选中表示不记录，默认为选中。

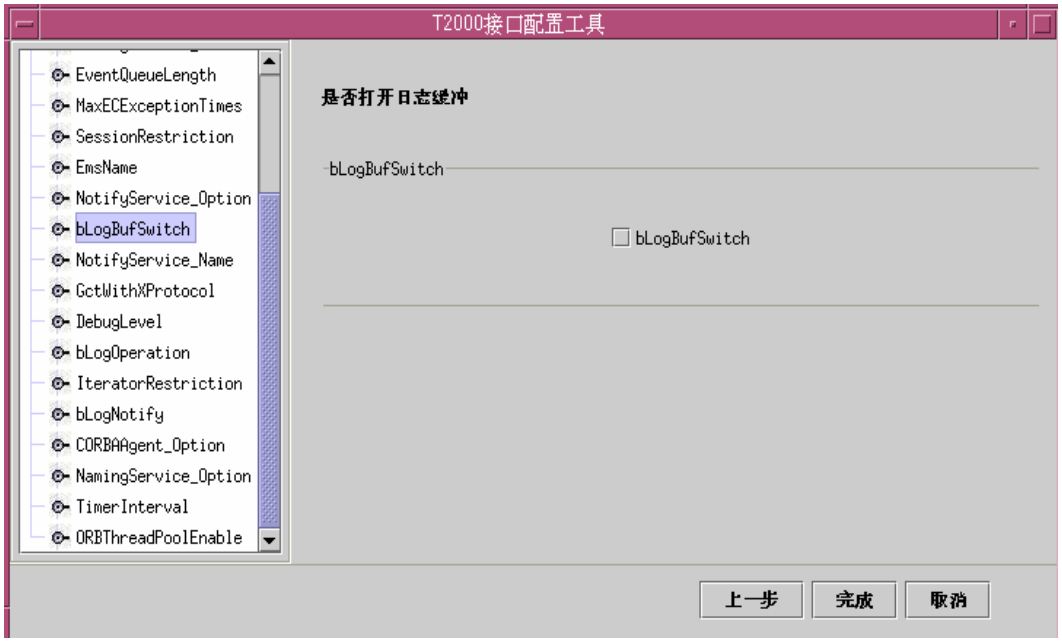
步骤 8 选择"bLogNotify"，出现如下界面。



说明

选择是否将上报通知写入日志。选中代表接口通知事件将记录在日志文件中，不选中表示不记录，默认为选中。

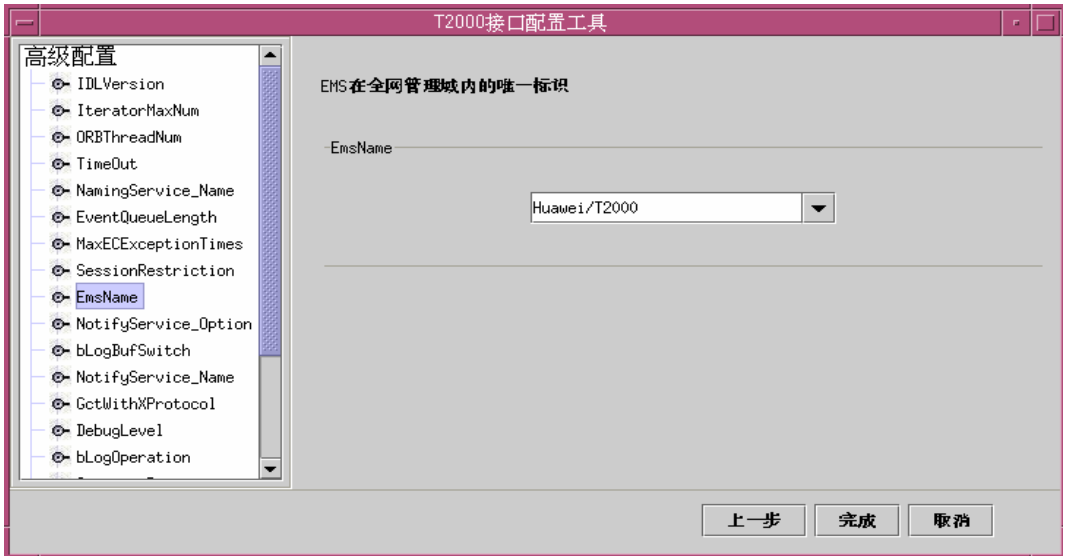
步骤 9 选择"bLogBufSwitch"，出现如下界面。



说明

选择是否实时写入日志。选中代表事件不会实时写入到日志文件中，不选中表示实时写入，默认为不选中。

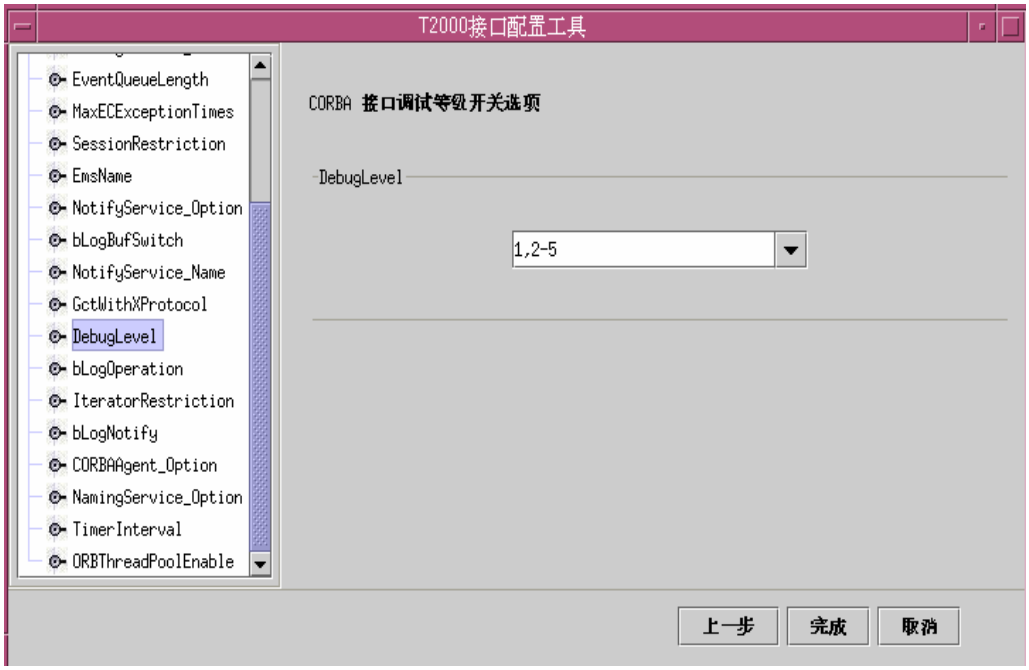
步骤 10 选择"EmsName"，出现如下界面。



说明

当网络规划中，需要配置多个 T2000 系统时，需要修改此配置参数，以确保不同的 T2000 系统的名字在 NMS 管理域内是唯一的。默认值是"Huawei/T2000"。

步骤 11 选择"Debug Level"，出现如下界面。



根据错误类型的不同，日志可以划分为 5 种级别：($IMIAP/corba.cfg 中有相应描述)

- 1st switch：CORBA platform error
- 2nd switch：Assert/Internal error
- 3rd switch：External error
- 4th switch：Function input/output
- 5th switch：Debug trace

可以根据需要，选择写入日志的错误类型。

日志级别的输入格式有比较严格的要求：

- 如果只选择单个级别且不为 1，则所选级别前所有位要用 0 补齐，用逗号分隔。例如：只选择级别 3，那么需要输入 0,0,3。
- 如果选择连续级别，可用横线连接。若不是从级别 1 开始，前面所有位仍要用 0 补齐，以逗号分隔。例如：选择级别 2,3,4，则可输入 0,2-4；选择级别 1,2,3,4,5，可以输入 1,2,3,4,5 或者 1-5。
- 如果选择不连续级别，空余位仍要用 0 补齐，以逗号分隔。例如：选择级别 1,3,5，则输入：1,0,3,0,5；选择级别 2,4，则输入：0,2,0,4。

步骤 12 单击<下一步>，配置 SSL 参数，使用默认值即可。

说明

如果在步骤四选中"不启用 SSLIOP 协议"，跳过此步，进入步骤十三。

步骤 13 单击<完成>，结束配置。

至此 CORBA 接口软件的配置就完成了。

----结束

更新 t2000_emsstart 脚本

步骤 1 下载如下 t2000_emsstart.sh 文件：


t2000_emsstart.sh

将文件重命名为 t2000_emsstart

步骤 2 上传 t2000_emsstart 脚本

本操作在主用服务器和备用服务器上均执行。

将 t2000_emsstart 脚本使用 ASCII 方式 FTP 到工作站上，替换 /opt/T2000App 目录下的同名文件。

步骤 3 修改 t2000_emsstart 脚本权限

本操作在主用服务器和备用服务器上均执行。

如果当前用户不是 root 用户，请切换到 root 用户，并执行如下命令：

```
#cd /opt/T2000App
#chmod +x *
#chown t2000 *
```

----结束

启用 T2000 网管系统

如果是 Veritas 热备份高可用性系统：
如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令启动主用服务器 T2000 网管系统：

```
#hagrp -online T2000App -sys 主用服务器主机名称
```

如果是 Veritas 温备份高可用性系统：

如果当前用户不是 t2000 用户，请切换到 t2000 用户并执行如下命令启动主用服务器 T2000 网管系统：

```
$cd $IMAP/bin
$./t2000server
```

如果是 SunCluster 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令启动主用服务器 T2000 网管系统：

```
#scswitch -Z -g T2000App
```

如果是 Watchman 热备份高可用性系统：

按照如下方式启动主用服务器 T2000 网管系统：

使用 Watchman 客户端登陆主用服务器，单击客户端的<启动应用>按键。

重新启动 T2000 CORBA 接口

建议按照以下步骤重新启动 CORBA 接口。

如果是 Veritas 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下步骤：

步骤 1 关闭主用服务器 T2000 网管系统。

```
#hagrp -offline T2000App -sys 主用服务器主机名称
```

步骤 2 启动主用服务器 T2000 网管系统。

```
#hagrp -online T2000App -sys 主用服务器主机名称
```

----结束

如果是 Veritas 温备份高可用性系统：

如果当前用户不是 t2000 用户，请切换到 t2000 用户并执行如下步骤：

步骤 1 关闭主用服务器 T2000 网管系统。

```
$cd $IMAP/bin
$./shutdownserver
```

步骤 2 启动主用服务器 T2000 网管系统。

```
$cd $IMAP/bin
$./t2000server
```

----结束

如果是 SunCluster 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下步骤：

步骤 1 步骤一：关闭主用服务器 T2000 网管系统。

```
#scswitch -F -g T2000App
```

步骤 2 步骤二：启动主用服务器 T2000 网管系统。

```
#scswitch -Z -g T2000App
```

----结束

如果是 Watchman 热备份高可用性系统：

步骤 1 步骤一：关闭主用服务器 T2000 网管系统。

使用 Watchman 客户端登陆主用服务器，单击客户端的<关闭应用>按键。

步骤 2 步骤二：启动主用服务器 T2000 网管系统。

使用 Watchman 客户端登陆主用服务器，单击客户端的<启动应用>按键。

----结束

禁用 T2000 CORBA 接口

如果在 T2000 系统运行过程中不再需要提供 CORBA 接口功能，建议按照下面步骤来关闭主用服务器 T2000 CORBA 接口功能。

步骤 1 关闭主用服务器 T2000 网管系统。

Veritas 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令：

```
#hagrp -offline T2000App -sys 主用服务器主机名称
```

Veritas 温备份高可用性系统：

如果当前用户不是 t2000 用户，请切换到 t2000 用户并执行如下命令：

```
$cd $IMAP/bin
$./shutdownserver
```

SunCluster 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令：

```
#scswitch -F -g T2000App
```

Watchman 热备份高可用性系统：

使用 Watchman 客户端登陆主用服务器，单击客户端的<关闭应用>按键。

步骤 2 通过 T2000 配置工具关闭 CORBA 接口，如下图：

说明

如何启动 T2000 配置工具请参看 5.2.1.1 使用 T2000 配置工具对 CORBA 接口进行配置部分。

步骤 3 在主用服务器启动 T2000 网管系统。

如果是 Veritas 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令：

```
#hagrp -online T2000App -sys 主用服务器主机名称
```

如果是 Veritas 温备份高可用性系统：

如果当前用户不是 t2000 用户，请切换到 t2000 用户并执行如下命令：

```
$cd $IMAP/bin
$./t2000server
```

如果是 SunCluster 热备份高可用性系统：

如果当前用户不是 root 用户，请切换到 root 用户并执行如下命令：

```
#scswitch -Z -g T2000App
```

如果是 Watchman 热备份高可用性系统：

按照如下方式启动主用服务器 T2000 网管系统：

使用 Watchman 客户端登陆主用服务器，单击客户端的<启动应用>按键。

说明

备机禁止 T2000 CORBA 接口过程与上述过程一致，这里操作过程略。

----结束

5.4 附录

5.4.1 License 项与 CORBA 的关系

T2000 License 有两个控制项用于控制 CORBA 接口软件，它们分别是：

- SupportCorbaIFBase ：控制 CORBA 对外提供基本功能。
- SupportCorbaIFAdv ：控制 CORBA 对外提供高级功能，高级功能是依赖于基本功能的，如果 License 不支持基本功能，即使有支持高级功能的 License，CORBA 接口也是无法使用的。

说明

如果 License 中控制项的值为 1，表示 License 支持此项，为 0 代表不支持此项，例如：SupportCorbaIFBase=1 表示此 License 支持 CORBA 对外提供基本功能，SupportCorbaIFBase=0 则表示此 License 不支持 CORBA 对外提供基本功能。

用户需要根据自身的需要购买相应的 License，CORBA 模块与 License 项的对应关系如表 5-1：

表5-1 CORBA 模块与 License 项的对应关系表

CORBA 模块	简介	与 License 项对应关系
核心模块	提供基本的存量,告警,性能管理	SupportCorbaIFBase=1
MSTP 模块	提供 MSTP 相关的功能	
SNC 模块	提供 SDH 和 WDM 领域 SNC 管理功能	SupportCorbaIFAdv=1
GCT 模块	提供界面直通相关的功能	
控制平面模块	提供控制平面相关的功能	
以太网流域管理模块	提供流域的管理功能	

CORBA 的高级功能除了需要 T2000 License 中 SupportCorbaIFBase 和 SupportCorbaIFAdv 两个控制项支持外还需要其他控制项进行支持，其关系如表 5-2：

表5-2 CORBA 高级功能模块与 T2000 License 支持项的对应关系表

CORBA 高级功能模块	T2000 License 支持项
SNC 模块	SupportCorbaIFBase = 1 SupportCorbaIFAdv = 1
	SDH 领域 SupportSDHEndToEnd = 1
	WDM 领域 SupportWaveEndToEnd = 1
GTC 模块	SupportCorbaIFBase = 1 SupportCorbaIFAdv = 1
控制平面模块	SupportCorbaIFBase = 1 SupportCorbaIFAdv = 1 SupportAson = 1
以太网流域管理模块	SupportCorbaIFBase = 1 SupportCorbaIFAdv = 1 SupportIPEndToEnd = 1 SupportSDHEndToEnd = 1

6 维护操作

关于本章

本章描述内容如下表所示。

标题	内容
6.1 维护条件说明	T2000 CORBA 接口维护条件介绍。
6.2 日常维护操作	T2000 CORBA 接口日常维护操作介绍。
6.3 常见问题处理	T2000 CORBA 接口常见问题处理方法介绍。

6.1 维护条件说明

T2000 CORBA 接口的正常运行依赖于以下条件：

维护操作人员具有相关经验

- 熟悉 PC 机、Sun 工作站、Windows 以及 Solaris 的基本操作。
- 对 CORBA 技术有一定的理解。
- 了解 TMN（Telecommunication Management Network）电信管理网基本概念，熟悉 T2000 系统组网的基本结构。
- 能熟练完成 T2000 CORBA 接口的配置以及启停等常规维护操作。

说明

工作站维护请参见供应商提供的维护手册。

T2000 Server 运行正常

T2000 Server 的各进程（mdp、ss、toposvr、ems）已经启动成功且运行正常。可能通过察看 T2000 系统的监控系统（Sysmonitor）来判断 mdp、ss、toposvr、ems 是否已经启动且动行正常。

6.2 日常维护操作

下面对 T2000 CORBA 接口的日常维护操作进行集中说明。建议以下维护操作每周进行一次。

检查 CORBA 服务运行状态

- T2000 单机环境下，用户可以登录 T2000 系统监控系统（Sysmonitor）的客户端，检查名字服务(Naming_Service)和通知服务 (Notify_Service)进程运行是否正常。
- T2000 双机环境下，用户可以登录双机系统监控系统的客户端，检查名字服务(Naming_Service)和通知服务 (Notify_Service)进程运行是否正常。

备份 T2000 CORBA 接口相关日志

T2000 CORBA 接口的日志信息存放在$IMAP/log/corba*.txt 中，为避免生成的日志大量占用系统磁盘空间，需要用户定期对这些日志文件进行备份。这些日志文件中记录了 T2000 CORBA 接口的运行信息，以及上层网管通过 T2000 CORBA 接口所进行的操作的信息，因此当 T2000 CORBA 接口运行出现错误时，用户可以通过这些日志文件来快速定位问题。T2000 CORBA 接口的日志信息，通过手工拷贝到指定目录的方式来进行备份。

6.3 常见问题处理

6.3.1 启动 T2000 CORBA 接口失败应该怎么处理

启动 T2000 CORBA 接口失败一般都是因为配置文件配置不正确导致的。当出现启动错误时我们应该逐步的检查配置文件，确认所有配置项都正确配置。

建议按照以下步骤解决问题：

步骤 1 查询系统当前的 IP 地址，并参考"5.2.2 启动 T2000 CORBA 接口"检查 T2000 CORBA 接口配置文件中配置的 IP 地址是否和当前 IP 地址一致（接口相关的配置文件包括 corba.cfg、ems.cfg、Notify_Service.cfg 以及 Naming_Service.cfg）

步骤 2 检查是否已经删除了$IMAP/persistence.ns 和$IMAP/ns.ior 文件

步骤 3 检查是否已经删除了$IMAP/sysmoni.cfg 中包含 Naming_Service、Notify_Service 的两行数据

步骤 4 检查是否已经正确的打开了配置文件$IMAP/ems.cfg 中 T2000 CORBA 接口的模块加载控制开关

步骤 5 检查 T2000 License 文件是否支持 T2000 CORBA 接口功能。

----结束

6.3.2 如何判断 T2000 已经成功打开 CORBA 接口

如果日志文件$IMAP/log/corbasyslog*.txt（"*"表示生成该文件的时间戳，根据这个时间戳可判断该日志是何时生成的。）中有如下信息：

```
===[CorbaAgentFacade::Init] Corba Agent is running! ===
```

说明 T2000 CORBA 接口已经正常启动。

如果$IMAP/log/目录下没有 corbasyslog*.txt 文件或者该文件中没有包含上面的信息则表明 T2000 CORBA 接口没有被加载或者 T2000 CORBA 接口加载失败。

6.3.3 T2000 CORBA 接口和上层网管对接需要做哪些准备工作

建议按照以下步骤操作：

步骤 1 按照"5.2.2 启动 T2000 CORBA 接口"启动 T2000 CORBA 接口；

步骤 2 在运行 T2000 的主机的 hosts 文件中加入上层网管的主机名称和 IP 地址的映射；

步骤 3 通过 T2000 网管客户端新建网管用户并分配网络管理员权限，供上层网管接入 T2000 CORBA 接口时使用。

----结束

6.3.4 如何判断 T2000 网管是否支持 CORBA 接口功能

通过如下两种方法可以判断 T2000 网管是否支持 CORBA 接口功能：

- 查看 T2000 的 License 文件是否支持 CORBA 接口功能：如果 T2000 的 License 支持 CORBA 接口功能，那么 License 文件中的 License 控制项：SupportCorbaIFBase 的值必须为 1，SupportCorbaIFAdv 为可选 CORBA 功能控制项，其值可能为 0 或者为 1。
- 登陆 T2000 客户端查看 License 文件：在 T2000 客户端界面主菜单中选择帮助>关于>注册信息，可以查看当前 T2000 License 是否支持 CORBA 接口功能。

7 接口信息模型

关于本章

本章描述内容如下表所示。

标题	内容
7.1 背景知识	T2000 CORBA 接口信息模型背景知识介绍。
7.2 功能实现声明	T2000 CORBA 接口功能实现声明介绍。
7.3 CORBA 服务说明	T2000 CORBA 接口服务说明。
7.4 通知事件格式	T2000 CORBA 接口通知事件格式介绍。
7.5 层速率说明	T2000 CORBA 接口层速率说明。
7.6 传输参数说明	T2000 CORBA 接口传输参数说明。
7.7 AdditionalInfo 使用说明	T2000 CORBA 接口 AdditionalInfo 使用说明。
7.8 接口限制和约束说明	T2000 CORBA 接口限制和约束说明。
7.9 对象命名规则	T2000 CORBA 接口对象命名规则介绍。
7.10 非功能互通性声明	T2000 CORBA 接口非功能互通性声明。

在阅读本章内容时，请同时参考下面的 TMF 建议文档。

- TMF 513 V3.0
- TMF 608 V3.0
- TMF 814 V3.0
- TMF 814A V3.0

7.1 背景知识

T2000 CORBA 接口是基于电信管理论坛（TMF）制定的 MTNM 系列标准开发的标准北向接口， 同时根据实际需要对标准接口进行了适当的扩展。

T2000 CORBA 接口参考了 TMF MTNM 的 TMF MTNM V3.0 标准，同时 T2000 CORBA 增加了部分私有接口。

本章中对于 T2000 CORBA 接口信息模型的描述，将参考 TMF 814A 标准的模板格式来进行描述。

7.1.1 管理器对象定义

高层网管如果要通过 T2000 CORBA 接口获取数据或交互请求，必须首先获取到相应的 CORBA 接口管理对象，即 CORBA 对象，然后通过该管理对象提供的方法来交互请求。目前，T2000 CORBA 接口中定义的管理对象以及对应的管理对象的名字如表 7-1 所示。

表7-1 管理对象定义与管理对象名字关系

管理对象定义	管理对象名字
EMSMgr_I	EMS
TopoMgr_I	TopoManagement
ProtectionMgr_I	Protection
FlowDomainMgr_I	FlowdomainManagement
MaintenanceMgr_I	Maintenance
GuiCutThroughMgr_I	GuiCutThrough
TrafficDescriptorMgr_I	CORBA_MSTP_TD
ManagedElementMgr_I	ManagedElement
EquipmentInventoryMgr_I	EquipmentInventory
MultiLayerSubnetworkMgr_I	MultiLayerSubnetwork
EncapsulationLayerLinkMgr_I	ELLManagement
PerformanceManagementMgr_I	PerformanceManagement
HW_controlPlaneMgr_I	ControlPlane
HW_MSTPInventoryMgr_I	CORBA_MSTP_INV
HW_MSTPProtectionMgr_I	CORBA_MSTP_PRO
HW_MSTPServiceMgr_I	CORBA_MSTP_SVC

说明

带"HW"前缀的管理对象表示华为在 TMF MTNM 标准的基础上扩展的管理对象。部分管理对象的名字可能和 TMF MTNM 标准不同，例如 TrafficDescriptorMgr_I 等，实际使用时以表格中的名字为准。上层网管也可以直接通过 **emsSession::EmsSession_I::getSupportedManagers** 方法获取 CORBA 接口支持的管理对象的名字列表。

下面通过一段简单的示例代码，说明上层网管获取管理对象的过程。

步骤 1 上层网管通过名字服务与 T2000 CORBA 接口建立连接。

说明

详细的步骤参见 7.3.2 名字服务编程示例。这里只列出建立连接的步骤。

```
emsSession::EmsSession_I_var emsSessionInterface;
    try{
EmsSessionFactory->getEmsSession( "userName", "userPassword", NmsSession,
emsSessionInterface.out() );
}catch( CORBA::Exception& ex )
    {
        //异常处理
    }
```

步骤 2 获取 T2000 CORBA 接口支持的管理对象名字列表。

```
emsSession::EmsSession_I::managerNames_T_var supportedMgrNameList = NULL;
    emsSessionInterface->getSupportedManagers( supportedMgrNameList );
```

步骤 3 根据管理对象的名字获取管理对象。

```
common::Common_I_ptr pMgr =
emsSessionInterface->getManager("ManagedElement");
    managedElement::ManagedElement_I_var meMgr =
        managedElement::ManagedElement_I::_narrow( pMgr );
    //调用 managedElement::ManagedElement_I 对象的方法进行请求交互
    meMgr->getAllManagedElements( ….);
```

----结束

7.1.2 实体对象定义

T2000 CORBA 接口管理的实体对象与 T2000 网管中定义的实体对象之间的对应关系如表 7-2 所示，对象命名及样例请参考 7.9 对象命名规则。

表7-2 实体对象定义关系

序号	对象名称	对象定义
1	EMS	表示 EMS 识别信息。
2	Subnetwork	表示多层子网信息，T2000 中没有与之对应的实体对象。
3	TopoSubnetwork	表示拓扑子网信息，和 T2000 中的拓扑子网相对应。
4	ProtectionSubnetwork	表示保护子网信息，和 T2000 中的保护子网相对应。

序号	对象名称	对象定义
5	SubnetworkConnection	表示连接终端点之间的逻辑连接，和 T2000 中的 SDH 路径、ASON 电路以及 WDM 中的 OCh 路径、OCh Client 路径等实体对象相对应。
6	ManagedElement	表示网元信息，和 T2000 中的 SDH 网元以及光网元相对应。
7	TopologicalLink	表示 EMS 内管理的网络拓扑连接信息，和 T2000 中的光纤以及复用段路径对象相对应。
8	EPGP	表示设备保护的保护组信息，和 T2000 中的设备保护组相对应。
9	PTP	表示物理终结点信息。Physical Termination Point (PTP)是 G.805/G.803 中 CPs 和 G.805/G.803 TCPs 的结合体，表示独立的物理点，和 T2000 中端口的概念对应。
10	CTP	表示连接终结点信息。Connection Termination Point (CTP)是 ITU G.805/G.803 中 Connection Point (CP) 和 G.805/G.803 中 Termination Connection Point (TCP)的结合体，和 T2000 中通道的概念对应。
11	TrafficDescriptor	表示 ATM 流量描述符信息。
12	EquipmentHolder	表示网元内机柜、子架和槽位信息，与 T2000 中的机柜、子架和槽位对应。
13	Equipment	表示单板信息，和 T2000 中的单板对象相对应。
14	ProtectionGroup	表示 MS 层保护的保护组信息，和 T2000 中的复用段保护组相对应。
15	WDM ProtectionGroup	表示 WDM 领域端口保护的保护组信息。
16	VirtualBridge	表示以太网内虚拟网桥的数据结构信息，和 T2000 中的虚拟网桥对象相对应。
17	VLAN	表示以太网内虚拟局域网的数据结构信息，和 T2000 中的 VLAN 转发过滤表相对应。
18	Ethernet Service	表示描述以太网内 EPL/EVPL/EPLn/EVPLn 单站业务信息, 和 T2000 以太网单站业务相对应。
19	ATM Service	表示描述 ATM 内 VPC/VCC 的业务信息，和 T2000 中的 ATM 交叉连接相对应。
20	ATM ProtectGroup	表示定义 ATM 保护组对象实体的数据结构信息，和 T2000 中的 ATM 保护组相对应。
21	QoS Rule	表示以太网内 QoS 规则数据的结构信息，和 T2000 的 QoS 规则对象相对应。
22	Flow	表示以太网内流的结构信息，和 T2000 的流对象相对应。
23	Flow Domain	用于表示以太网内流域的结构信息，T2000 中没有与之对应的实体对象。
24	FlowDomainFragment	表示以太网内 FDFr 的结构信息，对应 T2000 内以太端到端业务对象。

序号	对象名称	对象定义
25	EncapsulationLayerLink	表示以太网内 ELL 的结构信息，对应 T2000 内 TrunkLink 对象。
26	LinkAggregationGroup	表示链路聚合组的结构信息，和 T2000 的链路聚合组相对应。
27	RPRNode	表示以太网内 RPR 节点结构信息，和 T2000 的 RPR 节点相对应。
28	Routing Area	表示路由域结构信息，T2000 中没有与之对应的管理实体。
29	SNPPLink	表示控制平面内 SNPPLink 结构信息，对应于 T2000 的 TELink 对象。

7.2 功能实现声明

本节介绍了 T2000 CORBA 接口各功能模块的实现情况。包括如下内容。

- 说明
- 常用参数
- 功能模块

7.2.1 说明

T2000 CORBA 接口依照 TMF814 V3.0 定义的接口来实现。本章对比 TMF814 V3.0 的接口定义，列出了 T2000 CORBA 接口对于标准 CORBA 接口所定义各模块的实现情况以及华为实现的私有接口实现情况，具体从以下几个方面描述：

- 数据类型：列出模块涉及的主要对象数据类型，如：PTP、CTP、ManagedElement 等。
- 接口：对于模块操作的具体支持情况进行简要说明。
- 通知：列举模块涉及的通知信息。

数据类型

对各功能模块的数据类型的说明包括以下内容：

- 属性名称（Attributes Name）
- 属性设置者（Set by）：可选值为 EMS/NMS/E，其中 E 表示该属性既可以由 NMS 设置，也可以由 T2000 来设置其值。
- 属性设置修改标识（Set When and Set How）：可选值为 C/A/L，其中 C 表示设置该属性仅仅在对象创建的时候设置；A 表示该属性在任何时候均可以进行设置和修改；L 表示该属性在其生命周期内是可以设置修改的。
- 属性值格式（Format）：可选值为 FREE/FIXED/VALUE LIST，其中 FREE 表示对该属性值没有特殊限制；FIXED 表示该属性值的值域是严格限制的，只能是有限的几种值；VALUE LIST 则表示该属性值是一个序列。
- 特别说明（Clarification Needed）

接口

对各功能模块的接口的说明将包括以下内容：

- 操作名称（Operation）
- 是否为必选项（Status）：表示该操作是否是必须支持的，可选值为 M/O；M 表示该操作是必须支持的操作，O 表示该操作是可以选择支持的操作。
- 是否支持（Support）：表示 T2000 CORBA 接口是否支持该操作，可选值为 Y/N/C；Y 表示支持，N 表示不支持，C 表示有条件的支持。
- 异常类型（Exception），T2000 CORBA 接口支持 TMF814 建议定义的异常集合，具体如下：
  - EXCPT_NOT_IMPLEMENTED ：该功能没有实现
  - EXCPT_INTERNAL_ERROR：EMS 内部错误
  - EXCPT_INVALID_INPUT：输入非法
  - EXCPT_ENTITY_NOT_FOUND：实体不存在
  - EXCPT_NE_COMM_LOSS：与网元通讯失败
  - EXCPT_USERLABEL_IN_USE：userLabel 已经被使用
  - EXCPT_TOO_MANY_OPEN_ITERATORS：迭代器数目达到最大值
  - EXCPT_ACCESS_DENIED：无权访问
  - EXCPT_UNABLE_TO_COMPLY：EMS 不能执行该请求
  - EXCPT_OBJECT_IN_USE：对象已经被使用
  - EXCPT_CAPACITY_EXCEEDED：超出容量许可范围
  - EXCPT_TIMESLOT_IN_USE：时隙被占用
  - EXCPT_UNSUPPORTED_ROUTING_CONSTRAINTS：EMS 不支持该路由限制
  - EXCPT_PROTECTION_EFFORT_NOT_MET：NMS 要求的 SNC 保护不能实现
  - EXCPT_NOT_IN_VALID_STATE：NMS 试图删除处于激活状态的 SNC
  - EXCPT_TP_INVALID_ENDPOINT：指定的 TP 不存在或者不能够被创建
- 功能简述（Function Description）
- 备注（Comments）

通知

说明相应功能模块中各对象所支持的通知类型。

7.2.2 常用参数

下面介绍各模块数据类型及接口使用过程中将用到的标识符及通用迭代器对象。

标识符

- userLabel：用户标识符，由 NMS 指定。
- nativeEmsName：本地名称，即显示在 T2000 侧界面的对象名称。

- owner：对象所有者名称，由 NMS 指定。

通用迭代器对象

T2000 CORBA 接口完全支持 TMF814 中定义的各种迭代器（Iterator），提供了一整套迭代器的管理方法，能保证迭代器资源在任何情况下（正常和异常）都能得到正确的释放，防止内存泄漏。具体使用可参考 7.10.3 迭代器使用

7.2.3 Common 模块

数据类型

无。

接口

Common_I 接口描述如表 7-3 所示。

表7-3 Common_I 接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getCapabilities	M	Y	无	查询 manager 支持的功能	-
setNativeEMSName	O	Y	EXCPT_NOT_IMPLEMENTED	设置对象的 nativeEMSName 属性	具体支持的对象为： ME、SNC、TopologicalLink
setOwner	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR, EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	设置对象的 owner 属性	具体支持的对象为： EMS、Equipment、PTP、 SNC、TopologicalLink、PGP、ME、MultiLayerSubnetwork
setUserLabel	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	设置对象的 userLabel 属性	具体支持的对象为： EMS、Equipment、PTP、 SNC、TopologicalLink、PGP、ME、MultiLayerSubnetwork

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_ATTRIBUTE_VALUE_CHANGE	Y	对象属性改变通知	EMS、PTP、Equipment、SNC、TopologicalLink、PGP、ME、MultiLayerSubnetwork 的 nativeEMSName、userLabel 或 owner 属性	-

7.2.4 EmsMgr 模块

数据类型

EmsMgr 模块的数据类型描述如表 7-4 所示。

表7-4 EMS_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	E	A	FREE	-
nativeEMSName	EMS	A	FREE	不支持 NMS 对此属性值的修改操作
owner	NMS	L	FREE	-
emsVersion	EMS	C	FREE	-
type	EMS	C	FREE	-
additionalInfo	EMS	C	FREE	T2000 CORBA 接口在该属性中填充附加字段信息，具体可参考附加字段使用说明章节

接口

EMSMgr_I 接口描述如表 7-5 所示。

表7-5 EMSMgr_I 模块的接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getAllEMSAndMEActiveAlarms	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_TOO_MANY_OPEN_ITERATORS	查询 EMS 自身和所有网元的当前告警和 TCA 信息	-
getAllEMSSystemActiveAlarms	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_TOO_MANY_OPEN_ITERATORS	查询 EMS 自身的当前告警	-
getAllTopLevelSubnetworkNames	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_TOO_MANY_OPEN_ITERATORS	查询 EMS 的所有顶层子网名称	目前 T2000 只支持一个子网，子网类型为 MESH
getAllTopLevelSubnetworks	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_TOO_MANY_OPEN_ITERATORS	查询 EMS 的所有顶层子网信息	-
getAllTopLevelTopologicalLinkNames	O	N	无	查询 EMS 的所有顶层链路名称信息	不支持该接口
getAllTopLevelTopologicalLinks	O	N	无	查询 EMS 的所有顶层链路信息	不支持该接口
getEMS	M	Y	EXCPT_INTERNAL_ERROR	查询 EMS 信息	-
getTopLevelTopologicalLink	M	N	EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询 EMS 指定的顶层链路信息	不支持该接口
getObjectRootAlarms	O	N	EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询对象指定的一段时间内的根源告警	不支持该接口

通知

无。

7.2.5 Ems Session 模块

数据类型

无。

接口

EmsSession_I 接口描述如表 7-6 所示。

表7-6 EmsSession_I 模块的接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getEventChannel	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_ACCESS_DENIED	获取事件通道	-
getManager	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、	获取指定 Manager 的对象引用	-
getSupportedManagers	M	Y	EXCPT_INTERNAL_ERROR	获取 EMS 支持的 Manager 对象的名字列表	-

通知

无。

7.2.6 EmsSessionFactory 模块

数据类型

无。

接口

EmsSessionFactory_I 接口描述如表 7-7 所示。

表7-7 EmsSessionFactory_I 接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getEmsSession	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ACCESS_DENIED	获取 Ems Session 对象的对象引用，建立 Session 连接	-

通知

无。

7.2.7 Equipment 模块

数据类型

Equipment 模块的数据类型描述如表 7-8 所示。

表7-8 Equipment_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
Name	EMS	C	FREE	-
userLabel	E	A	FREE	-
nativeEMSName	EMS	A	FREE	-
Owner	NMS	A	FREE	-
alarmReportingIndicator	EMS	C	FIXED	由于粒度差异，T2000 CORBA 接口不支持此属性，该值固定为 true
serviceState	EMS	A	FIXED	表示单板的工作状态
expectedEquipmentObjectType	EMS	C	FIXED	表示网管上配置的单板类型
installedEquipmentObjectType	EMS	L	FIXED	表示网元上实际所插的单板类型
installedPartNumber	EMS	C	FREE	空值
installedVersion	EMS	C	FREE	单板软件版本
installedSerialNumber	EMS	C	FREE	单板序列号
additionalInfo	EMS	A	FREE	T2000 CORBA 接口在该属性中填充附加字段信息，具体可参考附加字段使用说明章节

EquipmentHolder 的数据类型描述如表 7-9 所示。

表7-9 EquipmentHolder_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	E	A	FREE	-
nativeEMSName	EMS	A	FREE	-
owner	NMS	L	FREE	-
alarmReportingIndicator	EMS	C	FIXED	由于粒度差异，T2000 CORBA 接口不支持此属性，该值固定为 true
holderType	EMS	C	FIXED	支持的类型包括：rack/shelf/slot
expectedOrInstalledEquipment	EMS	L	FIXED	安装或配置的单板名称，仅对 slot 有效
acceptableEquipmentTypeList	EMS	A	FREE	支持的单板类型列表，仅对 slot 有效；对其它类型，此属性没有意义
holderState	EMS	A	FIXED	EquipmentHolder 的状态，仅对 slot 有效； 对其它类型，此属性没有意义
additionalInfo	EMS	A	FREE	T2000 CORBA 接口在该属性中填充附加字段信息，具体可参考附加字段使用说明章节

Shelf 的物理位置信息数据类型描述如表 7-10 所示

表7-10 Shelf_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	L	FREE	子架的名称
equipmentRoomName	EMS	A	FREE	子架所在的机房名称
cabinetName	EMS	A	FREE	子架所属机柜的名称
numbering	NMS	L	FREE	子架的整机编号方式
locationOfCabinet	EMS	A	FIXED	子架在机柜中位置
memo	EMS	C	FREE	子架的备注信息

Cabinet 的物理位置信息数据类型描述如表 7-11 所示

表7-11 Cabinet_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	L	FREE	机柜的名称
equipmentRoomName	EMS	L	FREE	机柜所属的机房
containedShelfList	EMS	L	FREE	机柜包含的子架列表
type	NMS	L	FREE	机柜的类型
height	EMS	C	FIXED	机柜的高度（单位 mm）
width	EMS	C	FIXED	机柜的宽度（单位 mm）
depth	EMS	C	FIXED	机柜的深度（单位 mm）
voltage	EMS	L	FIXED	机柜的电压（单位 V）
powerBoxType	EMS	L	FIXED	机柜的电源和类型
memo	EMS	L	FREE	机柜的备注信息

EquipmentRoom 的物理位置数据类型描述如表 7-12 所示

表7-12 EquipmentRoom_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	L	FREE	机房名称
containedCabinet	EMS	L	FREE	包含的机柜列表
containedNMManager	EMS	L	FREE	包含的网管列表
country	EMS	L	FREE	所在的国家
province	EMS	L	FREE	所在的省份
city	EMS	L	FREE	所在的城市
site	EMS	L	FREE	站点名称
location	EMS	L	FREE	机房的位置信息
cableArrange	EMS	L	FREE	走线方式
defendStaticFloor	EMS	L	FREE	是否含有静电防护板

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
floorHeight	EMS	C	FIXED	静电防护板的高度
memo	EMS	L	FREE	备注信息

接口

EquipmentInventoryMgr_I 的接口描述如表 7-13 所示。

表7-13 EquipmentInventoryMgr_I 的接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getAllEquipment	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取网元或子架中包含的所有子设备	-
getAllEquipmentNames	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取网元或子架中包含的所有的子设备的名字	-
getAllSupportedPTPNames	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取单板支持的所有 PTP 的名字	-
getAllSupportedPTPs	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取单板支持的所有 PTP	-
getAllSupportingEquipment	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取包含指定 PTP 的单板列表	-
getAllSupportingEquipmentNames	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取包含指定 PTP 的单板的名称列表	-

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getContainedEquipment	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	获取子架直接包含的子设备	-
getEquipment	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	获取指定的子架或单板	-
provisionEquipment	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_UNABLE_TO_COMPLY	创建单板	-
setAlarmReportingOff	O	C	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_UNABLE_TO_COMPLY	关闭设备的告警上报开关	由于设置粒度存在差异，对此操作的支持有一定限制；不能对单个类型的告警进行设置
setAlarmReportingOn	O	C	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_UNABLE_TO_COMPLY	打开设备的告警上报开关	支持设置操作，但是不上报状态改变通知
unprovisionEquipment	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_UNABLE_TO_COMPLY	删除单板	-
getPhysicalLocationInfo	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询物理信息	-
getEquipmentStaticInfo	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询单板的静态信息	-

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_OBJECT_CREATION	Y	对象创建通知	Equipment、EquipmentHolder（shelf）	-
NT_OBJECT_DELETION	Y	对象删除通知	Equipment、EquipmentHolder（shelf）	-

7.2.8 GuiCutThrough 模块

数据类型

GCTProfileInfo 的数据类型描述如表 7-14 所示。

表7-14 GCTProfileInfo_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
serverLaunchCapability	EMS	C	FIXED	支持 Server 启动模式
gctHostname	EMS	C	FREE	IP 地址或者是主机名
emsGctPlatform	EMS	C	FIXED	-
guiCutThroughDataList	EMS	C	参见表 7-15	-

GuiCutThroughData 的数据类型描述如表 7-15 所示。

表7-15 GuiCutThroughData_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
gctScope	EMS	C	FIXED	-
gctContext	EMS	C	FIXED	-
gctCommand	EMS	C	FREE	-
additionalInfo	EMS	C	FREE	-

接口

GuiCutThrouthMgr_I 的接口描述如表 7-16 所示。

表7-16 GuiCutThrouthMgr_I 的接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
destroyGCT	O	N	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	关闭指定地址上的 GCT	-
getGCTProfileInfo	M	N	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询 GCT 的详细信息	-
launchGCT	O	N	EXCPT_CAPACITY_EXCEEDED、EXCPT_INTERNAL_ERROR	在指定地址上启动 GCT	-

通知

无。

7.2.9 MaintenanceOperations 模块

数据类型

CurrentMaintenanceOperation 的数据类型描述如表 7-17 所示。

表7-17 CurrentMaintenanceOperation_T 的类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
tpName	EMS	C	FIXED	可以为 PTP 或 CTP
maintenanceOperation	EMS	C	FIXED	支持的维护操作包括：设备环回（外环回）、终端环回（内环回）、AIS 告警插入、RDI 告警插入
layerRate	EMS	C	FIXED	-
additionalInfo	EMS	A	FREE	-

PRBSTestParameter 的数据类型定义描述如表 7-18 所示

表7-18 PRBSTestParameter_T 的类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
tpName	EMS	L	FIXED	测试的端口名称
testDuration	E	C	FIXED	测试周期
sampleGranularity	E	C	FIXED	采样周期
testType	E	C	FIXED	测试类型
accumulatingIndicator	E	C	FIXED	是否累计误码值

PRBSTestResult 的数据类型定义描述如表 7-19 所示

表7-19 PRBSTestResult_T 的类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
testPara	E	C	FREE	可参考 PRBSTestParameter_T 描述
startTime	E	C	FIXED	测试开始时间
sampleResultList	E	L	FIXED	测试采样结果
totalBitError	E	L	FIXED	测试比特误码总数
realDuration	E	L	FIXED	实际测试周期

接口

MaintenanceMgr_I 的接口描述如表 7-20 所示。

表7-20 MaintenanceMgr_I 的接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getActiveMaintenanceOperations	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取所有当前正在执行的维护操作	该接口不支持 DWDM 设备

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
performMaintenanceOperation	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_UNABLE_TO_COMPLY	执行指定的维护操作，执行方式可以为设置维护命令或清除维护命令	SDH 网元支持的维护操作包括：设备环回、终端环回、AIS 告警插入、RDI 告警插入。波分网元仅支持波分 PTP 的环回设置操作
enablePRBSTest	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_NE_COMM_LOSS	启动 PRBS 测试	该接口不支持 DWDM 设备
disablePRBSTest	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_NE_COMM_LOSS	停止 PRBS 测试	该接口不支持 DWDM 设备
getPRBSTestResult	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_NE_COMM_LOSS	查询 PRBS 测试结果	该接口不支持 DWDM 设备

通知

无。

7.2.10 ManagedElement 模块

数据类型

Managed Element 模块的数据类型描述如表 7-21 所示。

表7-21 Managed Element 模块的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	E	A	FREE	-
nativeEMSName	EMS	A	FREE	-
owner	NMS	L	FREE	-
location	EMS	C	FREE	-
version	EMS	C	FREE	网元版本
productName	EMS	C	FREE	网元名称
communicationState	EMS	L	FIXED	网元通信状态
emsInSyncState	EMS	L	FIXED	EMS 和网元的数据同步状态.
supportRates	EMS	A	VALUE LIST	网元支持的交叉连接速率列表
additionalInfo	EMS	A	FREE	T2000 CORBA 接口在该属性中填充附加字段信息，具体可参考附加字段使用说明章节

接口

无。

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_ATTRIBUTE_VALUE_CHANGE	Y	对象属性改变通知	userLabel、owner、supportedRates	-
NT_STATE_CHANGE	Y	对象状态改变通知	communicationState、emsInSyncState	-

7.2.11 ManagedElementMgr 模块

数据类型

无。

接口

ManagedElementMgr_I 接口描述如表 7-22 所示。

表7-22 ManagedElementMgr_I 模块的接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getAllActiveAlarms	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	获取网元所有当前告警和 TCA 信息	-
getAllCrossConnections	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取网元内包含的所有交叉连接的特征信息（新增包括波分交叉连接的查询（含固定交叉连接和动态交叉连接））	-
getAllManagedElementNames	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_TOO_MANY_OPEN_ITERATORS	获取所有网元的名字；其中波分网元包含的交叉连接层速率有LR_Optical_Channel、LR_Optical_Multiplex_Section、LR_OPTICAL_SECTION、LR_DIGITAL_SIGNAL_RATE 的信息。	-
getAllManagedElements	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_TOO_MANY_OPEN_ITERATORS	获取所有网元的特征信息	-

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getAllPTPs	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取网元内包含的所有PTP 的特征信息	-
getAllPTPNames	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取网元内包含的所有PTP 的名称	-
getContainedCurrentTPNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取 TP 包含的所有当前可用的 CTP 的名称	该接口不支持 DWDM 设备
getContainedCurrentTPs	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取 TP 包含的所有当前可用的 CTP 的特征信息	该接口不支持 DWDM 设备
getContainedInUseTPNames	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取 TP 包含的所有正在使用的 CTP 的名称	被 SNC 使用或被终结； 该接口不支持 DWDM 设备

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getContainedInUseTPs	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取 TP 包含的所有正在使用的 CTP 的特征信息	该接口不支持 DWDM 设备
getContainedPotentialTPNames	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取 TP 包含的所有潜在的 CTP 的名字	-
getContainedPotentialTPs	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	获取 TP 包含的所有潜在的 CTP 的特征信息	-
getContainingSubnetworkNames	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	获取网元所属子网的名称	-
getContainingTPNames	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	获取包含指定 CTP 的 TP 的名称列表	该接口不支持 DWDM 设备
getContainingTPs	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	获取包含指定 CTP 的所有 TP 的特征信息	该接口不支持 DWDM 设备

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getManagedElement	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	获取指定网元的特征信息	-
getTP	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	获取指定 TP 的特征信息	-
setTPData	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_ENTITY_NOT_FOUND	设置 TP 参数	目前支持告警屏蔽设置、开销字节设置、波分端口可调波长、TCM 检视、激光器状态、自协商模式功能
getActiveAlarms	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定对象上的当前告警	指定对象可以是：ME、TP、Equipment 以及 EquipmentHolder
checkActiveAlarms	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、	核对当前告警	-
createCrossConnections	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、	创建交叉连接	仅支持 SDH 的交叉连接
activeCrossConnections	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、	激活交叉连接	仅支持 SDH 的交叉连接
deactiveCrossConnections	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、	去激活交叉连接	仅支持 SDH 的交叉连接

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
deleteCrossConnections	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、	删除交叉连接	仅支持 SDH 的交叉连接
getAllAlarmsByFTP	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、	查询 EMS 和所有网元内当前的所有告警，并将结果通过 FTP 方式传输	-
getNEStaticInfo	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、	查询网元静态信息	-

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_OBJECT_CREATION	Y	对象创建通知	Managed Element、PTP	-
NT_OBJECT_DELETION	Y	对象删除通知	Managed Element	-
NT_ATTRIBUTE_VALUE_CHANGE	Y	对象属性改变通知	Managed Element、PTP	支持 Managed Element、PTP 部分属性改变（边界点属性改变）通知。
NT_STATE_CHANGE	Y	对象状态改变通知	Managed Element	支持 Managed Element 状态（通信状态以及数据同步状态）改变通知。

7.2.12 MTNM Version 模块

数据类型

无。

接口

Version_I 的接口描述如表 7-23 所示。

表7-23 Version_I 的接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getVersion	M	Y	无	获取 EMS 支持的 IDL 的版本信息	T200 CORBA 接口支持的 IDL 版本为 V3.0

通知

无。

7.2.13 MultiLayerSubnetwork 模块

数据类型

MultiLayerSubnetwork 模块的数据类型描述如表 7-24 所示。

表7-24 MultiLayerSubnetwork_T 模块的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	E	A	FREE	如果由 EMS 创建，缺省值为 rootsn
nativeEMSName	EMS	A	FREE	如果由 EMS 创建，缺省值为 rootsn
owner	NMS	L	FREE	-
subnetworkType	EMS	C	FIXED	只支持 MESH 型子网
supportedRates	EMS	A	VALUE LIST	-
additionalInfo	EMS	A	FREE	-

接口

MultiLayerSubnetworkMgr_I 的接口描述如表 7-25 所示。

表7-25 MultiLayerSubnetworkMgr_I 的接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
activateSNC	O	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND	激活指定的 SNC	-
checkValidSNC	O	Y	EXCPT_NOT_IMPLEMENTED EXCPT_INTERNAL_ERROR EXCPT_INVALID_INPUT EXCPT_OBJECT_IN_USE EXCPT_ENTITY_NOT_FOUND EXCPT_TIMESLOT_IN_USE EXCPT_UNABLE_TO_COMPLY EXCPT_NE_COMM_LOSS EXCPT_NOT_IN_VALID_STATE	判断是否能够创建 SNC	-
createAndActivateSNC	M	C	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND、 EXCPT_UNABLE_TO_COMPLY、 EXCPT_UNSUPPORTED_ROUTING_CONSTRAINTS	创建并激活指定的 SNC：先创建 SNC，然后再激活该 SNC	接口限制说明参见 7.8 接口限制和约束说明章节 新增支持 GE 速率 Client 路径创建
createSNC	O	C		创建 SNC	接口限制说明参见 7.8 接口限制和约束说明章节 新增支持 GE 速率 Client 路径创建
deactivateAndDeleteSNC	M	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND	去激活并删除指定的 SNC	-
deactivateSNC	O	Y		去激活指定的 SNC	-
deleteSNC	O	Y		删除指定的 SNC	-

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getAllEdgePoints	M	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND、 EXCPT_TOO_MANY_OPEN_ITERATORS	查询指定的子网内所有的边界点，可以用来创建相关的SNC（非VC4级别的SNC）	-
getAllEdgePointNames	M	Y		查询指定子网内所有边界点的名称	-
getAllManagedElements	M	Y		查询指定子网内所有的网元	-
getAllManagedElementNames	M	Y		查询指定子网内所有网元的名称	-
getAllSubnetworkConnections	M	Y		查询指定子网内所有的SNC	层速率支持：2、3、4、5、7、8、11、13、14、15、16、17、18、29、40、87。5（E1）和11（VC12）、7（E3）和13（VC3）是同级的； 新增支持OCH路径，GE速率Client路径查询
getAllSubnetworkConnectionNames	M	Y		查询指定子网内所有SNC的名称	层速率支持：2、3、4、5、7、8、11、13、14、15、16、17、18、29、40、87。5（E1）和11（VC12）、7（E3）和13（VC3）是同级的； 新增支持OCH路径，GE速率Client路径查询

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getAllSubnetworkConnectionsWithTP	M	Y		指定一个 CTP 或者 PTP，查询以该 TP 为源、宿或经过该 TP 的 SNC	层速率支持：2、3、4、5、7、8、11、13、14、15、16、17、18、29、40、87。5（E1）和 11（VC12）、7（E3）和 13（VC3）是同级的； 新增支持 OCH 路径，GE 速率 Client 路径查询
getAllSubnetworkConnectionNamesWithTP	M	Y		指定一个 CTP 或者 PTP，查询以该 TP 为源、宿或经过该 TP 的 SNC 的名称	层速率支持：2、3、4、5、7、8、11、13、14、15、16、17、18、29、40、87。5（E1）和 11（VC12）、7（E3）和 13（VC3）是同级的； 新增支持 OCH 路径，GE 速率 Client 路径查询
getAllTopologicalLinks	O	Y		查询子网内所有的 TopoLogicalLink	-
getAllTopologicalLinkNames	O	Y		查询子网内所有的 TopoLogicalLink 的名称	-
getAllTPPoolNames	O	N	EXCPT_NOT_IMPLEMENTED	查询子网内所有的 TPPool 的名称	不支持该接口
getAllTPPools	O	N	EXCPT_NOT_IMPLEMENTED	查询子网内所有的 TPPool 的特征信息	不支持该接口
getAssociatedTP	O	N	EXCPT_NOT_IMPLEMENTED	查询相关保护关系的 TP，主要是用来创建 SNC 时指定关系	不支持该接口
getTPGroupingRelationships	O	N	EXCPT_NOT_IMPLEMENTED	查询 TPPool 相关的 TP 或者 TP 相关的 TPPool	不支持该接口

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getMultiLayerSubnetwork	M	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND	通过子网名称，查询子网的详细信息	-
getRoute	O	C	EXCPT_NOT_IMPLEMENTED、 EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND	查询指定 SNC 的路由详细信息	
getRoutes	O	C	EXCPT_NOT_IMPLEMENTED、 EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND	查询指定多条 SNC 的路由详细信息	
getSNC	M	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND	通过 SNC 名称，查询 SNC 的详细信息	
getSNCsByUserLabel	M	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND	查询指定 UserLabel 的所有 SNC 的详细信息	
getTopologicalLink	M	Y		通过名称，查询 Topological 的详细信息	
getSNCs	O	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT	查询指定的多条 SNC 的详细信息	-

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getSNCsByEndObjectName	O	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND	通过指定 SNC 经过的终端点，查询满足条件的 SNC 的详细信息	-
getSNCsByNativeEmsName	O	Y	EXCPT_INTERNAL_ERROR	通过 nativeEMSName 查询相应的 SNC 的详细信息	层速率支持：2、3、4、5、7、8、11、13、14、15、16、17、18、29、40、87。5（E1）和 11（VC12）、7（E3）和 13（VC3）是同级的。
getAllInternalTopologicalLinks	O	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND、 EXCPT_TOO_MANY_OPEN_ITERATORS	查询光网元内部的拓扑连接	-
getAllInternalTopologicalLinkNames	O	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND、 EXCPT_TOO_MANY_OPEN_ITERATORS	查询光网元内部拓扑连接的名称	-
modifySNC	O	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND	修改 SNC	-
getRouteAndTopologicalLinks	O	Y	EXCPT_INTERNAL_ERROR、 EXCPT_INVALID_INPUT、 EXCPT_ENTITY_NOT_FOUND	查询 SNC 的路由和拓扑连接信息	-

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
setConjunctionSNC	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	设置关联 SNC	该接口只支持 ASON 设备；
swapSNC	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	SNC 切换	该接口只支持 SDH 电路；

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_OBJECT_CREATION	Y	对象创建通知	TopologicalLink、SNC	-
NT_OBJECT_DELETION	Y	对象删除通知	TopologicalLink、SNC	-
NT_ATTRIBUTE_VALUE_CHANGE	Y	对象属性改变通知	TopologicalLink、SNC、MultilayerSubnetwork	-
NT_STATE_CHANGE	Y	对象状态改变通知	SNC	-
NT_ROUTE_CHANGE	Y	对象路由改变通知	SNC	-

7.2.14 Performance 模块

数据类型

PMData 的数据类型描述如表 7-26 所示。

表7-26 PMData_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
tpName	EMS	C	Fixed	-
layerRate	EMS	C	Fixed	-
granularity	EMS	C	Fixed	-
retrievalTime	EMS	C	Fixed	-
pmMeasurementList	EMS	C	-	参见 PMMeasurement

PMMeasurement 的数据类型描述如表 7-27 所示。

表7-27 PMMeasurement_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
pmParameterName	EMS	C	VALUE LIST	-
pmLocation	EMS	C	Fixed	-
value	EMS	C	Fixed	-
unit	EMS	C	FREE	-
intervalStatus	EMS	C	Fixed	-

PMThresholdValue 的数据类型描述如表 7-28 所示。

表7-28 PMThresholdValue_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
pmParameterName	EMS	C	VALUE LIST	-
pmLocation	EMS	C	Fixed	-
thresholdType	EMS	C	Fixed	-
triggerFlag	EMS	C	Fixed	-
value	EMS	C	Fixed	-
unit	EMS	C	FREE	-

接口

PerformanceManagementMgr_I 的接口描述如表 7-29 所示。

表7-29 PerformanceManagementMgr_I 的接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
clearPMData	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	请求 EMS 清除（重置）指定的测量点的 PM 寄存器	-
disablePMData	M	C	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	禁止性能数据采集	对 ME 级别完全支持；但只支持对同一 TP 的 15 分钟和 24 小时性能监视的同时禁止（对主机软机版本为 4.x 的产品不区分 15 分钟和 24 小时性能）； 该接口对部分 DWDM 设备支持
enablePMData	M	C	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_CAPACITY_EXCEEDED	使能性能数据采集	对 ME 级别完全支持；但只支持对同一 TP 15 分钟和 24 小时性能监视的同时使能； 该接口对部分 DWDM 设备支持
disableTCA	O	N	EXCPT_NOT_IMPLEMENTED	禁止性能越限告警	不支持该接口

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
enableTCA	O	N	EXCPT_NOT_IMPLEMENTED	使能越门限告警	不支持该接口
getAllCurrentPMData	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_TOO_MANY_OPEN_ITERATORS	查询所有当前性能数据	以太网 RMON 和 ATM 当前性能的查询
getHistoryPMData	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询历史性能数据	以太网 RMON 和 ATM 历史性能的查询
getHoldingTime	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询 EMS 能够保存多长时间的性能数据记录	保存时长：24 小时性能：144 小时；15 分钟性能：4 小时
getMEPMcapabilities	M	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询网元性能管理能力	-
getTCATPParameter	M	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询 TP 的 TCA 参数	-
setTCATPParameter	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	设置 TP 的 TCA 参数	-
getPMState	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询性能监视的状态	-

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
createPMCollectionTask	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	创建性能采集任务	
deletePMCollectionTasks	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	删除性能采集任务	
getPMCollectionTasks	O	Y	EXCPT_NOT_IMPLEMENTED、EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询性能采集任务	输入为空则默认返回全部任务

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_TCA	Y	TCA 事件通知	TCA 通知	-
NT_FILE_TRANSFER_STATUS	Y	文件传输状态通知	文件传输状态通知	查询历史性能数据接口上报文件传输状态

7.2.15 Protection 模块

数据类型

ProtectionGroup 的数据类型描述如表 7-30 所示。

表7-30 ProtectionGroup_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	E	A	FREE	初始值为空
nativeEMSName	EMS	A	FREE	初始值为空
owner	NMS	L	FREE	初始值为空
protectionGroupType	EMS	C	FIXED	支持的保护组类型有： PGT_MSP_1PLUS1； PGT_MSP_1FORN； PGT_2F_BLSR； PGT_4F_BLSR
protectionSchemeState	EMS	A	FIXED	保护策略
reversionMode	EMS	C	FIXED	恢复模式
rate	EMS	C	FIXED	-
pgpTPList	EMS	C	FIXED	-
pgpParameters	EMS	C	FIXED	保护组参数列表，各种保护类型参数列表有一定差别。包括有：SwitchMode、wtrTime、HoldOffTime、LODNumSwitches、LODDuration、SPRINGProtocol、SPRINGNodeId、SwitchPosition 以及 nonPre-EmptibleTraffic
additionalInfo	EMS	A	FREE	-

ProtectionSubnetwork 的数据类型描述如表 7-31 所示。

表7-31 ProtectionSubnetwork_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	E	A	FREE	初始值为空
nativeEMSName	EMS	A	FREE	初始值为空
owner	NMS	L	FREE	初始值为空
layerRate	EMS	C	FIXED	-
psnType	EMS	A	FIXED	保护子网类型
neIDList	EMS	A	FIXED	网元 ID 列表
psnLinks	EMS	A	FIXED	-
additionalInfo	EMS	A	FREE	-

EprotectGroup 的数据类型描述如表 7-32 所示

表7-32 EprotectGroup_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	E	A	FREE	-
nativeEMSName	EMS	A	FREE	-
owner	NMS	L	FREE	-
eProtectionGroupType	EMS	C	FIXED	支持的设备保护组类型为 1_FOR_N
protectionSchemeState	EMS	A	FIXED	保护策略
reversionMode	EMS	C	FIXED	恢复模式
protectedList	EMS	C	FIXED	-
protectingList	EMS	C	FIXED	-
ePgpParameters	EMS	C	FIXED	该参数固定为空
additionalInfo	EMS	L	FIXED	-

WDMprotectGroup 的数据类型描述如表 7-33 所示

表7-33 WDMprotectGroup_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	E	A	FREE	-
nativeEMSName	EMS	A	FREE	-
owner	NMS	L	FREE	-
protectionGroupType	EMS	C	FIXED	支持的波分端口保护组类型为：1_PLUS_1、1_FOR_N
protectionSchemeState	EMS	A	FIXED	保护策略
reversionMode	EMS	C	FIXED	恢复模式
pgpTPList	EMS	C	FIXED	-
pgpParameters	EMS	C	FIXED	保护组参数列表，恢复模式下为恢复等待时间 wtrTime
additionalInfo	EMS	C	FIXED	-

接口

ProtectionMgr_I 的接口描述如表 7-34 所示。

表7-34 ProtectionMgr_I 的接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getAllProtectionGroups	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定网元中所有的保护组	不包括单向MSP；该接口不支持DWDM 设备
getProtectionGroup	M	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	根据保护组名称查询该保护组的详细信息	该接口不支持DWDM 设备

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getAllNUTTPNames	O	N	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询指定保护组中所有"携带不可预占无保护额外业务"CTP 的名称	不支持该接口
getAllPreemptibleTPNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询指定保护组中所有"携带可预占额外业务"CTP 的名称	该接口不支持 DWDM 设备
getAllProtectedTPNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询指定保护组中所有"携带被保护业务"CTP 的名称	该接口不支持 DWDM 设备
retrieveSwitchData	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定保护组或者 SNCP 的当前倒换状态	该接口不支持 DWDM 设备
performProtectionCommand	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	在指定保护组或者 SNCP 上执行外部倒换命令	该接口不支持 DWDM 设备
getAdjacentTPs	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询保护 TP 的相关 TP	-

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
getAllEProtectionGroups	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询所有的设备保护组	-
getEProtectionGroup	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定的设备保护组	-
retrieveESwitchData	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询设备保护组的倒换信息	-
getAllProtectionSubnetworks	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询所有保护子网	-
getAllWDMProtectionGroups	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询所有的波分保护组	-
getWDMProtectionGroup	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定的波分保护组	-

操作名称	是否为必选项	是否支持	异常类型	功能简述	备注
retrieveWDMSwitchData	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询波分保护组的倒换数据	-
performWDMProtectionCommand	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_UNABLE_TO_COMPLY	对波分保护组执行外部倒换命令	-

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_OBJECT_CREATION	Y	对象创建通知	Protection Group	-
NT_OBJECT_DELETION	Y	对象删除通知	Protection Group	-
NT_ATTRIBUTE_VALUE_CHANGE	Y	对象属性改变通知	Protection Group	-
NT_STATE_CHANGE	Y	对象状态改变通知	Protection Group	-
NT_PROTECTION_SWITCH	Y	保护倒换通知		-
NT_WDMPROTECTION_SWITCH	Y	保护倒换通知		-

7.2.16 Session 模块

数据类型

无。

接口

Session_I 的接口描述如表 7-35 所示。

表7-35 Session_I 模块的接口描述

操作名称	是否为必选项	是否支持	异常类型	功能简述
endSession	M	Y	终断 Session 连接	-
ping	M	Y	检测通信是否正常	用于判断通信连接状态的好坏。如果 ping 失败会释放 EMSSesssion 资源，断开连接。 默认调用周期为 30 秒，修改周期可参考系统安装运行章节的"配置选项说明"。

通知

无。

7.2.17 Subnetwork Connection 模块

数据类型

Crossconnection 的数据类型描述如表 7-36 所示。

表7-36 Crossconnection_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
active	EMS		FIXED	-
aEndNameList	E	该属性可以被 NMS 和 EMS 进行设置，NMS 中可以通过 CreateSNC、ActivateSNC、CreateAndActivateSNC、Deactivate 等接口进行设置	FIXED	-
zEndNameList	E	如上	FIXED	-
direction	E		FIXED	交叉连接支持单向
ccType	E		FIXED	支持的类型有： ST_SIMPLE； ST_ADD_DROP_A； ST_ADD_DROP_Z； ST_INTERCONNECT； ST_DOUBLE_INTERCONNECT； ST_DOUBLE_ADD_DROP；

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
				ST_OPEN_ADD_DROP； ST_EXPLICIT 创建时推荐使用 ST_SIMPLE 单向类型。
additionalInfo	EMS	A	FREE	T2000 CORBA 接口在该属性中填充附加字段信息，具体可参考附加字段使用说明章节

Subnetwork Connection 的数据类型描述如表 7-37 所示。

表7-37 SubnetworkConnection_T 的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	E	A	FREE	-
nativeEMSName	E	A	FREE	-
owner	NMS	L	FREE	-
sncState	EMS	A	FIXED	EMS 可支持 pending 状态，但是只支持在 pending 状态时相关资源也被认为是已被占用
direction	E	A	FIXED	-
rate	E	C	FIXED	-
staticProtection	E	C	FIXED	SDH 电路创建时不支持此参数，但是查询电路可以支持此参数，支持的保护级别为有： PREEMPTIBLE； UNPROTECTED； FULLY_PROTECTED； HIGHLY_PROTECTED； PARTIALLY_PROTECTED。 创建 ASON 电路中，该字段代表路径的保护能力。

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
sncType	E	C		支持的类型有： ST_SIMPLE； ST_ADD_DROP_Z； ST_ADD_DROP_A； ST_EXPLICIT
aEnd	E	C	FIXED	-
zEnd	E	C	FIXED	-
rerouteAllowed	E	C	FIXED	-
networkRouted	E	C	FIXED	指示路由是否必须在网络级别计算实现。 ASON 电路中该字段为 NR_YES； SDH 电路该字段为 NR_NO。
additionalInfo	EMS	A		T2000 CORBA 接口在该属性中填充附加字段信息，具体可参考附加字段使用说明章节

接口

无。

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_ATTRIBUTE_VALUE_CHANGE	Y	对象属性改变通知	支持对象的 userLabel、owner、aEnd、zEnd 属性	-
NT_STATE_CHANGE	Y	对象状态改变通知	SNC	-

7.2.18 Termination Point 模块

数据类型

Termination Point 模块的数据类型描述如表 7-38 所示。

表7-38 TerminationPoint_T 据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	E	A	FREE	-
nativeEMSName	E	A	FREE	-
owner	NMS	L	FREE	-
ingressTrafficDescriptorName	NMS	A	FIXED	不支持
egressTrafficDescriptorName	NMS	A	FIXED	不支持
type	EMS	C	FIXED	支持的类型有： TPT_PTP； TPT_CTP。
connectionState	EMS	A	FIXED	连接状态
tpMappingMode	E	A	FIXED	映射模式
direction	E	C	FIXED	方向
transmissionParams	E	A	FIXED	传输参数，该参数可参考层速率和传输参数列表章节
tpProtectionAssociation	EMS	A	FIXED	保护相关，支持的方式有： TPPA_PSR_RELATED； TPPA_NA。
edgePoint	EMS	A	FIXED	边界点，空闲的光口即认为是边界点。
additionalInfo	EMS	A	FREE	-

接口

无。

通知

无。

7.2.19 Topological Link 模块

数据类型

Topological Link 模块的数据类型描述如表 7-39 所示。

表7-39 TopologicalLink_T 模块的数据类型描述

属性名称	属性设置方式	属性设置时间或设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	E	A	FREE	-
nativeEMSName	E	A	FREE	-
owner	E	A	FREE	-
direction	EMS	C	FIXED	-
rate	EMS	C	FIXED	-
aEndTP	EMS	C	FIXED	-
zEndTP	EMS	C	FIXED	-
additionalInfo	EMS	A	FREE	-

接口

无。

通知

无。

7.2.20 HW_mstpInventory 模块

数据类型

HW_MSTPEndPoint_T 的数据类型描述如表 7-40 所示。

表7-40 HW_MSTPEndPoint_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	EMS	C	FREE	-
nativeEMSName	EMS	C	FREE	-
owner	EMS	C	FREE	-
direction	EMS	C	FIXED	-
type	EMS	C	FIXED	支持的类型有： HW_MEPT_NA； HW_MEPT_ATM； HW_MEPT_ATMTRUNK HW_MEPT_ETH； HW_MEPT_ETHTRUNK HW_MEPT_LP； HW_MEPT_RPR；
transmissionParams	E	C	FIXED	传输参数，该参数可参考层速率和传输参数列表章节
additionalInfo	EMS	A	FREE	-

HW_VirtualBridge_T 的数据类型描述如表 7-41 所示。

表7-41 HW_VirtualBridge_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	EMS	C	FREE	-
nativeEMSName	EMS	A	FREE	-
owner	EMS	C	FREE	-
logicalTPList	E	C	FIXED	-

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
additionalInfo	EMS	A	FREE	-

HW_VirtualLAN_T 的数据类型描述如表 7-42 所示。

表7-42 HW_VirtualLAN_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	EMS	C	FREE	-
nativeEMSName	EMS	C	FREE	-
owner	EMS	C	FREE	-
paraList	EMS	C	FIXED	-
forwardTPList	E	C	VALUE LIST	-
additionalInfo	EMS	A	FREE	-

HW_MSTPBindingPath_T 的数据类型描述如表 7-43 所示。

表7-43 HW_MSTPBindingPath_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
direction	E	C	FIXED	-
allPathList	E	C	VALUE LIST	所有绑定通道的名称列表
usedPathList	EMS	C	VALUE LIST	绑定中被使用的通道名列表
additionalInfo	EMS	A	FREE	-

HW_ForwardEndPoint_T 的数据类型描述如表 7-44 所示。

表7-44 HW_ForwardEndPoint_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
logicTPName	EMS	C	FREE	-
paraList	EMS	A	FIXED	-
additionalInfo	EMS	A	FREE	-

HW_QosRule_T 的数据类型描述如表 7-45 所示

表7-45 HW_QosRule_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	QoS 规则的名字
userLabel	E	A	FIXED	-
nativeEMSName	E	L	FREE	-
owner	EMS	L	FREE	-
type	EMS	C	FIXED	支持的 QoS 规则的类型有： HW_QT_NA； HW_QT_CAR； HW_QT_COS。
paraList	EMS	L	FIXED	QoS 规则支持的属性列表
additionalInfo	EMS	A	FREE	-

HW_Flow_T 的数据类型描述如表 7-46 所示

表7-46 HW_Flow_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	Flow 的名字
userLabel	E	A	FIXED	-
nativeEMSName	EMS	L	FREE	-
owner	E	L	FREE	-
qosRuleNames	EMS	L	FIXED	Flow 绑定的 QoS 规则列表

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
paraList	EMS	L	FIXED	Flow 的参数列表，根据不同情况，可包含以下参数： FlowType、 PortType、 PortID、 VlanID、 Priority、 vbID
additionalInfo	EMS	A	FREE	-

HW_LinkAggregationGroup_T 的数据类型描述如表 7-47 所示

表7-47 HW_LinkAggregationGroup_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	LAG 的名字
userLabel	E	A	FIXED	-
nativeEMSName	EMS	L	FREE	-
owner	E	L	FREE	-
mainPortName	EMS	C	FREE	链路聚合的主端口
branchPortNameList	EMS	A	FIXED	-
paraList	EMS	L	FREE	
additionalInfo	EMS	A	FREE	-

接口

HW_MSTPInventoryMgr_I 的接口描述如表 7-48 所示。

表7-48 HW_MSTPInventoryMgr_I 的接口描述

Operation	Status	Support	Exception	Function Description	Comments
getAllMstpEndPoints	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有的 MSTP 终端点	-
getAllMstpEndPointNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有的 MSTP 终端点名称	-
getMstpEndPoint	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定的 MSTP 终端点	-
setMstpEndPoint	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	设置指定的 MSTP 终端点	-
getAllVBs	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有的 VB	-
getAllVBNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有 VB 的名称	-
getVirtualBridge	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定的虚拟网桥	-
createVirtualBridge	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	创建虚拟网桥	-
deleteVirtualBridge	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	删除虚拟网桥	-
createVLAN	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	创建 VLAN	-

Operation	Status	Support	Exception	Function Description	Comments
deleteVLAN	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	删除 VLAN	-
getAllVLANs	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询 VB 内所有的 VLAN 信息	-
getAllVLANNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询 VB 内的 VLAN 的名称	-
getVLAN	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定的 VLAN	-
addVLANForwardPort	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	增加转发过滤端口	-
delVLANForwardPort	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	删除转发过滤端口	-
setVLANData	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	设置 VLAN 属性	-
getBindingPath	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询 VCTrunk 绑定通道	-
addBindingPath	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	增加 VCTrunk 绑定通道	-
delBindingPath	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	删除 VCTrunk 绑定通道	-
getLCASState	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询 LCAS 状态	-
setLCASState	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	设置 LCAS 状态	-

Operation	Status	Support	Exception	Function Description	Comments
getAllContainedInUseTPNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询 ATM 或 ATMTrunk 终端点包含的所有正在被使用 CTP 的名字	-
getCapabilities	O	Y	无		-
createQosRule	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	创建 QoS Rule	-
setQosRule	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	设置 QoS Rule	-
createFlow	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	创建流	-
getAllQosRules	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有的 QoS 规则	-
getAllQosRuleNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询网元内所有的 QoS 规则的名字	-
getQosRule	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询网元内指定的 QoS	-
deleteQosRule	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	删除 QoS 规则	-
getAllFlows	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有的流对象	-
getAllFlowNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有的流对象的名字	-
getFlow	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询网元内指定的流对象	-

Operation	Status	Support	Exception	Function Description	Comments
setFlow	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	设置流和 QoS 的绑定关系	-
deleteFlow	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	删除指定的流对象	-
createLinkAggregationGroup	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	创建链路聚合组	-
getAllLinkAggregationGroups	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询网元内所有的链路聚合组	-
getAllLinkAggregationGroupNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询网元内所有的链路聚合组名称	-
getLinkAggregationGroup	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定链路聚合组	-
getAvailablePortNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询单板内可用于链路聚合的端口	-
modifyLinkAggregationGroup	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	设置链路聚合组	-
deleteLinkAggregationGroup	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	删除指定的链路聚合组	-
setMstpEndPointShapingQueue	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	设置端口的 Shaping 队列	-
getMstpEndPointShapingQueue	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询端口的 Sharping 队列	-

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_OBJECT_CREATION	Y	对象创建通知	VB、VLAN、MSTPEndPoint、QoS、FLOW、LAG	-
NT_OBJECT_DELETION	Y	对象删除通知	VB、VLAN、MSTPEndPoint、QoS、FLOW、LAG	-
NT_ATTRIBUTE_VALUE_CHANGE	Y	对象属性改变通知	VB、VLAN、MSTPEndPoint、QoS、FLOW、LAG	-
NT_STATE_CHANGE	Y	对象状态改变通知	VCTrunk、LCAS	支持 VCTrunk 通道绑定改变通知；支持 LCAS 状态改变通知。

7.2.21 HW_mstpService 模块

数据类型

HW_ETHServiceTP_T 的数据类型描述如表 7-49 所示。

表7-49 HW_ETHServiceTP_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
vlanID	EMS	C	FIXED	vlanID，创建业务时指定
tunnel	EMS	L	FIXED	tunnel 标签，创建业务时指定
vc	EMS	L	FIXED	VC 标签，创建业务时指定
additionalInfo	EMS	A	FREE	-

HW_ETHServiceCreateData_T 的数据类型描述如表 7-50 所示。

表7-50 HW_ETHServiceCreateData_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
serviceType	EMS	C	FREE	业务类型，创建业务时指定
direction	EMS	C	FIXED	业务方向
aEndPoint	EMS	C	FIXED	业务的源端 TP
zEndPoint	EMS	C	FIXED	业务的宿端 TP
additionalInfo	EMS	A	FREE	-

HW_ETHService_T 的数据类型描述如表 7-51 所示。

表7-51 HW_ETHService_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	EMS	C	FREE	-
nativeEMSName	EMS	C	FREE	-
owner	EMS	C	FREE	-
serviceType	EMS	C	FIXED	
direction	EMS	C	FIXED	-
activeState	E	L	FIXED	业务是否激活标志
aEndPoint	EMS	C	FIXED	-
zEndPoint	EMS	C	FIXED	-
additionalInfo	EMS	A	FREE	T2000 CORBA 接口在该属性中填充附加字段信息，具体可参考附加字段使用说明章节

HW_ATMServiceTP_T 的数据类型描述如表 7-52 所示。

表7-52 HW_ATMServiceTP_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
trafficDescriptorName	EMS	C	FREE	流量描述符名称
bPC	EMS	C	FIXED	bPC 属性标识是否使能网络和用户使用的参数
additionalInfo	EMS	A	FREE	-

HW_ATMService_T 的数据类型描述如表 7-53 所示。

表7-53 HW_ATMService_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FIXED	保护类型
userLabel	EMS	C		-
nativeEMSName	EMS	C		-
owner	EMS			-
protectType	EMS	C	FIXED	保护类型
serviceType	EMS	C	FIXED	业务类型
spreadType	EMS	C	FIXED	传播类型
protectRole	EMS	C	FIXED	保护角色
aEndPoint	EMS	C	FIXED	业务源端 TP
zEndPoint	EMS	C	FIXED	业务宿端 TP
active	E	L	FIXED	业务是否激活标志
additionalInfo	EMS	A	FREE	-

HW_ATMServiceCreateData_T 的数据类型描述如表 7-54 所示。

表7-54 HW_ATMServiceCreateData_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
protectType	EMS	C	FIXED	保护类型
serviceType	EMS	C	FIXED	业务类型
spreadType	EMS	C	FIXED	传播类型
protectRole	EMS	C	FIXED	保护角色
aEndPoint	EMS	C	FIXED	业务源端 TP
zEndPoint	EMS	C	FIXED	业务宿端 TP
active	E	L	FIXED	业务是否激活标志
additionalInfo	EMS	A	FREE	-

接口

HW_MSTPServiceMgr_I 的接口描述如表 7-55 所示。

表7-55 HW_MSTPServiceMgr_I 的接口描述

Operation	Status	Support	Exception	Function Description	Comments
createEthService	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	创建以太网业务	-
deleteEthService	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	删除以太网业务	-
getAllEthService	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有的以太网业务	-
getEthService	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定的以太网业务	-
createAtmService	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	创建 ATM 业务	-

Operation	Status	Support	Exception	Function Description	Comments
deleteAtmService	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	删除 ATM 业务	-
getAllAtmService	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有 ATM 业务	-
getAtmService	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定的 ATM 业务	-
activateAtmService	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	激活 ATM 业务	-
deactivateAtmService	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	去激活 ATM 业务	-
getCapabilities	O	Y	无	-	-

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_OBJECT_CREATION	Y	对象创建通知	Ethernet Service、ATM Service	-
NT_OBJECT_DELETION	Y	对象删除通知	Ethernet Service、ATM Service	-
NT_STATE_CHANGE	Y	对象状态改变通知	ATM Service	ATM Service 的激活、去激活状态改变

7.2.22 HW_mstpProtect 模块

数据类型

HW_RPRNode_T 的数据类型描述如表 7-56 所示。

表7-56 HW_RPRNode_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	EMS	C	FREE	-
nativeEMSName	EMS	C	FREE	-
owner	EMS	C	FREE	-
nodeNo	EMS	C	FIXED	节点号
protocolEnabled	EMS	C	FIXED	协议是否使能状态
nodeParameters	E	C	FREE	RPR 节点参数，包含以下字段： holdOffTime； wtrTime； protectMode； restoreMode； slowTime； topoTimer。
additionalInfo	EMS	A	FREE	-

HW_RPRSwitchData_T 的数据类型描述如表 7-57 所示。

表7-57 HW_RPRSwitchData_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
nodeName	EMS	C	FREE	-
switchReason	EMS	C	FIXED	倒换原因
switchState	EMS	A	FIXED	倒换状态
switchPosition	EMS	C	FIXED	倒换方向

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
switchParameters	EMS	C	FREE	倒换参数，包含以下参数： switchPosition； switchState； protectType； SwitchCountTimes ProtectCountTime LastSwitchCommand
additionalInfo	EMS	A	FREE	-

HW_RPRTopoInfo_T 的数据类型描述如表 7-58 所示。

表7-58 HW_RPRTopoInfo_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
nodeName	EMS	C	FREE	-
topoParameters	EMS	C	FREE	Topo 信息
additionalInfo	EMS	A	FREE	-

HW_ATMServiceProtectPair_T 的数据类型描述如表 7-59 所示。

表7-59 HW_ATMServiceProtectPair_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
workServiceName	EMS	C	FREE	ATM 工作业务名称
protectServiceName	EMS	C	FREE	ATM 保护业务名称
monitorFlag	EMS	C	VALUE LIST	监视标志
additionalInfo	EMS	A	FREE	-

HW_ATMSingleEndSwitchPara_T 的数据类型描述如表 7-60 所示。

表7-60 HW_ATMSingleEndSwitchPara_T 的数据类型描述

属性名称	书信设置方式	属性设置时间和设置方法	属性值格式	特别说明
switchReason	EMS	C	FIXED	-
switchState	EMS	C	FIXED	-
additionalInfo	EMS	A	FREE	-

HW_ATMPGSwitchData_T 的数据类型描述如表 7-61 所示。

表7-61 HW_ATMPGSwitchData_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
pgName	EMS	C	FREE	ATM 保护组名称
protectType	EMS	C	FIXED	-
srcEndSwitchPara	EMS	C	FIXED	源端倒换数据
snkEndSwitchPara	EMS	C	FXIED	宿端倒换数据
additionalInfo	EMS	A	-	-

HW_ATMProtectGroup_T 的数据类型描述如表 7-62 所示。

表7-62 HW_ATMProtectGroup_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	ATM 工作业务名称
userLabel	EMS	C	FREE	ATM 保护业务名称
nativeEMSName	EMS	C	FREE	监视标志
owner	EMS	C	FREE	-
protectType	EMS	L	FIXED	保护类型
switchDirect	EMS	L	FIXED	倒换方向
useState	EMS	L	FIXED	保护组使用状态标志
srcEndPara	EMS	C	FIXED	保护组源端数据
snkEndPara	EMS	C	FIXED	保护组宿端数据
ppList	EMS	L	VALUE LIST	保护对列表
additionalInfo	EMS	A	FREE	-

HW_ATMPGSingEndPara_T 的数据类型描述如表 7-63 所示。

表7-63 HW_ATMPGSingEndPara_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
switchType	EMS	C	FIXED	倒换类型
reversionMode	EMS	C	FIXED	恢复模式
additionalInfo	EMS	A	FREE	-
holdOffTime	EMS	L	FIXED	倒换延迟
wtrTime	EMS	L	FIXED	等待恢复时间

接口

HW_MSTPProtectMgr_I 的接口描述如表 7-64 所示。

表7-64 HW_MSTPProtectMgr_I 的接口描述

Operation	Status	Support	Exception	Function Description	Comments
getAllRPRNode	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有的 RPR 节点	-
getRPRNode	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指的 RPR 节点	-
getRPRTopoPara	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询 RPR 拓扑信息	-

Operation	Status	Support	Exception	Function Description	Comments
retrieveRPRSwitchData	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询 RPR 节点倒换信息	-
performRPRProtectionCommand	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	执行 RPR 保护倒换命令	-
getAllAtmProtectGroup	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询网元内所有的 ATM 保护组	-
getAtmProtectGroup	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定的 ATM 保护组信息	-
retrieveAtmPGSwitchData	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询 ATM 保护倒换数据	-
performAtmPGProtectionCommand	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	执行 ATM 保护倒换外部命令	-
getCapabilities	O	Y	无		-

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_OBJECT_CREATION	Y	对象创建通知	ATM 保护组，RPR 节点	-
NT_OBJECT_DELETION	Y	对象删除通知	ATM 保护组，RPR 节点	-
NT_ATTRIBUTE_VALUE_CHANGE	Y	对象属性改变通知	ATM 保护组，RPR 节点	-
NT_STATE_CHANGE	Y	对象状态改变通知	ATM 保护组	-
NT_PROTECTION_SWITCH	Y	保护倒换通知	ATM 保护组、RPR 节点	支持 ATM 保护组、RPR 节点保护倒换通知上报。

7.2.23 trafficDescriptor 模块

数据类型

TrafficDescriptor_T 的数据类型描述如表 7-65 所示。

表7-65 TrafficDescriptor_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	-
userLabel	EMS	C	FREE	-
nativeEMSName	EMS	C	FREE	-
owner	EMS	C	FREE	-
serviceCategory	EMS	C	FIXED	流量描述符类别
trafficParameters	EMS	C	FREE	流量描述符参数
conformanceDefinition	EMS	C	FREE	-
activeState	E	L	FIXED	是否激活标志
additionalInfo	EMS	A	FREE	-

TrafficDescriptor_T 的数据类型描述如表 7-66 所示。

表7-66 TDCreateData _T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
userLabel	EMS	C	FREE	-
forceUniqueness	EMS	C	FREE	-
owner	EMS	C	FREE	-
serviceCategory	EMS	C	FREE	-
trafficParameters	EMS	C	FIXED	传输参数，该参数可参考 7.5 层速率说明和 7.6 传输参数说明。
conformanceDefinition	EMS	C	FREE	-
activeState	E	L	FIXED	-
additionalInfo	EMS	A	FREE	-

接口

TrafficDescriptorMgr_I 的接口描述如表 7-67 所示。

表7-67 TrafficDescriptorMgr_I 的接口描述

Operation	Status	Support	Exception	Function Description	Comments
HW_getAllTrafficDescriptors	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有的流量描述符	-
HW_getAllTrafficDescriptorNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、EXCPT_TOO_MANY_OPEN_ITERATORS	查询网元内所有的流量描述符名称	-

Operation	Status	Support	Exception	Function Description	Comments
getTrafficDescriptor	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询指定流量描述符的信息	-
HW_createTrafficDescriptor	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	创建流量描述符	-
deleteTrafficDescriptor	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	删除流量描述符	-
getAllTrafficDescriptors	O	N	EXCPT_NOT_IMPLEMENTED	查询网元内所有的流量描述符	该接口不支持
getAllTrafficDescriptorNames	O	N	EXCPT_NOT_IMPLEMENTED	查询网元内所有的流量描述符名称	该接口不支持
createTrafficDescriptor	O	N	EXCPT_NOT_IMPLEMENTED	创建流量描述符	该接口不支持
activateTrafficDescriptor	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	激活流量描述符	-
deactivateTrafficDescriptor	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	去激活流量描述符	-
getCapabilities	O	Y	无		-

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_OBJECT_CREATION	Y	对象创建通知	ATM 流量描述符	-
NT_OBJECT_DELETION	Y	对象删除通知	ATM 流量描述符	-
NT_STATE_CHANGE	Y	对象状态改变通知	ATM 流量描述符	ATM 流量描述符的激活、去激活状态改变

7.2.24 控制平面管理模块

数据类型

HW_SnppLink_T 的数据类型描述如表 7-68

表7-68 HW_SnppLink_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	SNPPLink 的名称
userLabel	EMS	C	FREE	-
owner	EMS	C	FREE	-
nativeEMSName	EMS	C	FREE	-
aEndSnppName	EMS	C	FIXED	A 端 SNPP 名称
zEndSnppName	EMS	C	FREE	Z 端 SNPP 名称
direction	EMS	L	FIXED	方向
rateList	EMS	C	FREE	速率列表
cost	EMS	C	FIXED	权重
protectType	EMS	L	FIXED	保护类型
linkCapacity	EMS	L	FIXED	链路容量
linkState	EMS	L	FIXED	链路状态
srlgIDList	EMS	L	FIXED	共享链路风险组列表
additionalInfo	EMS	L	FIXED	-

接口

HW_controlPlaneMgr_I 的接口描述如表 7-69 所示

表7-69 HW_controlPlaneMgr_I 的接口描述

Operation	Status	Support	Exception	Function Description	Comments
getAllRoutingAreaNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询所有的路由域名称	-
getAllRoutingNodeNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询路由域内所有的路由节点名称	-
getAllSnppLinks	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询路由域内所有的 SNPPLink	-
getAllSnppNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询路由节点包含的 SNPP 名称	-
getAllContainedSnpNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询 SNPP 中包含的 SNP 名称	-
setSRLG	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	设置共享链路风险组	-

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_OBJECT_CREATION	Y	对象创建通知	SNPPLink	-
NT_OBJECT_DELETION	Y	对象删除通知	SNPPLink	-
NT_ATTRIBUTE_VALUE_CHANGE	Y	对象属性改变通知	SNPPLink	-
NT_ASON_SOURCE_CHANGE	Y	对象状态改变通知	Routing Area	Routing Area 创建通知、路由节点创建通知；Routing Area 删除通知、路由节点删除

7.2.25 FlowDomain 模块

数据类型

FlowDomain_T 的数据类型描述如表 7-70 所示

表7-70 FlowDomain _T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	FlowDomain 的名称
userLabel	EMS	C	FREE	-
owner	EMS	C	FREE	-
nativeEMSName	EMS	C	FREE	-
transmissionParams	EMS	C	FIXED	不支持该参数
networkAccessDomain	EMS	C	FREE	不支持该参数
mfds	EMS	L	FIXED	不支持该参数
fdEdgeCPTPs	EMS	C	FREE	不支持该参数
fdInternalCPTPs	EMS	C	FIXED	不支持该参数
fDConnectivityState	EMS	L	FIXED	该参数固定为 CS_UNKNOWN
fdType	EMS	L	FIXED	流域类型，该参数固定为 FDT_NETWORK
additionalInfo	EMS	L	FIXED	-

FlowDomainFragment_T 的数据类型描述如表 7-71

表7-71 FlowDomainFragment_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	FDFr 的名称
userLabel	EMS	C	FREE	-
owner	EMS	C	FREE	-
nativeEMSName	EMS	C	FREE	-
transmissionParams	EMS	C	FIXED	传输参数，该参数可参考 7.5 层速率说明和 7.6 传输参数说明。
networkAccessDomain	EMS	C	FREE	不支持该参数
fixed	EMS	C	FREE	该参数固定为 false
administrativeState	EMS	L	FIXED	该字段固定为 AS_Unlocked
additionalInfo	EMS	L	FIXED	-

接口

FlowDomainMgr_I 的接口描述如表 7-72 所示

表7-72 FlowDomainMgr_I 的接口描述

Operation	Status	Support	Exception	Function Description	Comments
getAllFlowDomains	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询所有的流域	-
getAllFlowDomainNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	查询所有的流域名称	-

Operation	Status	Support	Exception	Function Description	Comments
getFlowDomain	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND	通过流域名称查询该名称的流域详细信息	-
getAllFDFrs	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	通过流域查 FDFr 信息	-
getAllFDFrNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	通过流域查 FDFr 名称	-
getFDFr	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	查询指定的 FDFr 信息	-
createFDFr	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	创建 FDFr	-
activateFDFr	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	激活指定 FDFr	-
deactivateFDFr	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	去激活指定的 FDFr	-
deleteFDFr	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	删除指定的 FDFr	-
getFDFrServerTrail	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	获得 FDFr 对应的服务层路径	-
getAllEthernetOAMPoint	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	取得指定以太业务所有维护点	-

Operation	Status	Support	Exception	Function Description	Comments
performEthernetOAMCommandt	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	对维护点执行维护命令	-
getFlowDomainsByUserLabel	O	N	EXCPT_NOT_IMPLEMENTED	查询指定流域信息	该接口不支持
getFDfromMFD	O	N	EXCPT_NOT_IMPLEMENTED	查询指定流域信息	该接口不支持
getTransmissionParams	O	N	EXCPT_NOT_IMPLEMENTED	查询指定流域的传输参数	该接口不支持
createFlowDomain	O	N	EXCPT_NOT_IMPLEMENTED	创建流域	该接口不支持
deleteFlowDomain	O	N	EXCPT_NOT_IMPLEMENTED	删除指定流域	该接口不支持
modifyFlowDomain	O	N	EXCPT_NOT_IMPLEMENTED	修改指定流域	该接口不支持
associateMFDsWithFlowDomain	O	N	EXCPT_NOT_IMPLEMENTED	联合指定流域	该接口不支持
deassociateMFDsWithFlowDomain	O	N	EXCPT_NOT_IMPLEMENTED	去联合指定流域	该接口不支持
associateCPTPsWithFlowDomain	O	N	EXCPT_NOT_IMPLEMENTED	联合指定流域边界点	该接口不支持
deassociateCPTPsWithFlowDomain	O	N	EXCPT_NOT_IMPLEMENTED	去联合指定流域边界点	该接口不支持
getAllMFDs	O	N	EXCPT_NOT_IMPLEMENTED	查询流域矩阵	该接口不支持
getAllMFDNames	O	N	EXCPT_NOT_IMPLEMENTED	查询流域矩阵的名字	该接口不支持
getMFD	O	N	EXCPT_NOT_IMPLEMENTED	查询指定的流域矩阵	该接口不支持
getAssigningMFD	O	N	EXCPT_NOT_IMPLEMENTED	查询指派的流域矩阵	该接口不支持
createMFD	O	N	EXCPT_NOT_IMPLEMENTED	创建流域矩阵	该接口不支持
deleteMFD	O	N	EXCPT_NOT_IMPLEMENTED	删除指定的流域矩阵	该接口不支持
modifyMFD	O	N	EXCPT_NOT_IMPLEMENTED	修改指定的流域矩阵	该接口不支持

Operation	Status	Support	Exception	Function Description	Comments
assignCPTPsToMFD	O	N	EXCPT_NOT_IMPLEMENTED	指派流域矩阵	该接口不支持
unassignCPTPsToMFD	O	N	EXCPT_NOT_IMPLEMENTED	去指派流域矩阵	该接口不支持
createFTP	O	N	EXCPT_NOT_IMPLEMENTED	创建 FTP	该接口不支持
deleteFTP	O	N	EXCPT_NOT_IMPLEMENTED	删除 FTP	该接口不支持
getAllCPTPs	O	N	EXCPT_NOT_IMPLEMENTED	查询所有流域矩阵边界点	该接口不支持
getAssignableCPTPs	O	N	EXCPT_NOT_IMPLEMENTED	查询流域矩阵可指派的边界点	该接口不支持
getFDFrWithTP	O	N	EXCPT_NOT_IMPLEMENTED	根据 TP 名称查询经过该 TP 的 FDFr	该接口不支持
getFDFrByUserLabel	O	N	EXCPT_NOT_IMPLEMENTED	根据 UserLabel 查询 FDFr	该接口不支持
modifyFDFr	O	N	EXCPT_NOT_IMPLEMENTED	修改指定的 FDFr	该接口不支持
addFPsToFDFr	O	N	EXCPT_NOT_IMPLEMENTED	增加指定的 FDFr 的流点	该接口不支持
removeFPsToFDFr	O	N	EXCPT_NOT_IMPLEMENTED	减少指定的 FDFr 的流点	该接口不支持
validateTMDAssignmentToMFD	O	N	EXCPT_NOT_IMPLEMENTED	验证流域矩阵属性	该接口不支持

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_OBJECT_CREATION	Y	对象创建通知	FDFr	-
NT_OBJECT_DELETION	Y	对象删除通知	FDFr	-
NT_ATTRIBUTE_VALUE_CHANGE	Y	对象属性改变通知	FDFr	-
NT_STATE_CHANGE	Y	对象状态改变通知	FDFr	FDFr 激活、去激活

7.2.26 EncapsulationLayerLink 模块

数据类型

EncapsulationLayerLink_T 的数据类型描述如表 7-73 所示

表7-73 EncapsulationLayerLink_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	ELL 的名称
userLabel	EMS	C	FREE	-
owner	EMS	C	FREE	-
nativeEMSName	EMS	C	FREE	-
type	EMS	C	FIXED	只支持 LT_POINT_TO_POINT 的 ELL 类型。
transmissionParams	EMS	C	FIXED	传输参数，该参数可参考 7.5 层速率说明和 7.6 传输参数说明。
rate	EMS	L	FIXED	该字段固定为"LR_Encapsulation"
networkAccessDomain	EMS	C	FREE	该字段不支持
endTPs	EMS	C	FIXED	-
route	EMS	L	FIXED	服务层路由
segment	EMS	L	FIXED	是否未终结
routeGroups	EMS	L	FIXED	该字段不支持
additionalInfo	EMS	L	FIXED	T2000 CORBA 接口在该属性中填充附加字段信息，具体可参考附加字段使用说明章节

接口

EncapsulationLayerLinkMgr_I 的接口描述如表 7-74 所示

表7-74 EncapsulationLayerLinkMgr_I 的接口描述

Operation	Status	Support	Exception	Function Description	Comments
createELLink	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、	创建 ELL	-
getAllELLinks	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、	查询所有的 ELL	-
getAllELLinkNames	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、	查询所有 ELL 的名称	-
getELLink	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、	查询指定 ELL	-
activateELLink	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、	激活指定 ELL	-
deactivateELLink	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	去激活指定 ELL	-
deleteELLink	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	删除指定 ELL	-
increaseBandwidthOfELLink	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	增加指定 ELL 的带宽	-
decreaseBandwidthOfELLink	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	减少指定 ELL 的带宽	-
setELLinkLCASState	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT	设置指定 ELL 的 LCAS 状态	-
getELLinkWithTP	O	N	EXCPT_NOT_IMPLEMENTED	根据 TP 查询 ELL	该接口不支持
getConnectingELL	O	N	EXCPT_NOT_IMPLEMENTED	查询与 TP 端口相连的 ELL	该接口不支持

Operation	Status	Support	Exception	Function Description	Comments
getELLinkWithSncOrTl	O	N	EXCPT_NOT_IMPLEMENTED	根据 SNC 或逻辑拓朴接接查询 ELL	该接口不支持
getAllELLinksWithMeOrFd	O	N	EXCPT_NOT_IMPLEMENTED	根据网元 Me 或流域 Fd 查询指定 ELL	该接口不支持
getServerSNCsAndTLs	O	N	EXCPT_NOT_IMPLEMENTED	查询指定 ELL 的服务层路径	该接口不支持
getTransmissionParams	O	N	EXCPT_NOT_IMPLEMENTED	查询指定 ELL 的传输参数	该接口不支持
modifyELLink	O	N	EXCPT_NOT_IMPLEMENTED	修改指定 ELL	该接口不支持

通知

支持以下通知：

通知名称	是否支持	通知简述	支持对象	备注
NT_OBJECT_CREATION	Y	对象创建通知	ELL	-
NT_OBJECT_DELETION	Y	对象删除通知	ELL	-
NT_ATTRIBUTE_VALUE_CHANGE	Y	对象属性改变通知	ELL	-
NT_STATE_CHANGE	Y	对象状态改变通知	ELL	ELL 激活、去激活

7.2.27 TopoMgr 模块

数据类型

Node_T 的数据类型描述如表 7-75 所示

表7-75 Node_T 的数据类型描述

属性名称	属性设置方式	属性设置时间和设置方法	属性值格式	特别说明
name	EMS	C	FREE	Node 的名称
nativeEMSName	EMS	C	FREE	-
nodeType	EMS	C	FREE	-
position	EMS	C	FREE	坐标位置
parent	EMS	C	FIXED	该节点的父拓扑节点
additionalInfo	EMS	L	FIXED	-

接口

TopoMgr_I 的接口描述如表 7-76 所示

表7-76 TopoMgr_I 的接口描述

Operation	Status	Support	Exception	Function Description	Comments
getTopoSubnetworkViewInfo	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、	查询网管拓扑视图中的网元和子网信息，主要是坐标信息。	-
getProtectSubnetworkViewInfo	O	Y	EXCPT_INTERNAL_ERROR、EXCPT_INVALID_INPUT、EXCPT_ENTITY_NOT_FOUND、	查询网管保护子网视图中的网元信息，主要是坐标信息。	-

通知

无

7.3 CORBA 服务说明

7.3.1 概述

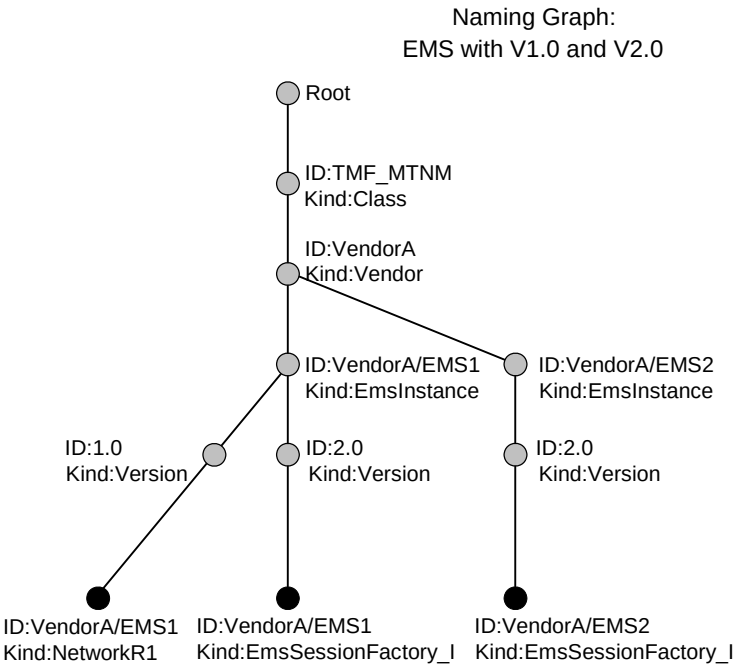
T2000 CORBA 接口需要使用标准的 CORBA 名字服务和通知服务。通过名字服务，T2000 为 NMS 正确的访问 T2000 CORBA 接口提供了唯一的入口。T2000 和 NMS 的对接支持多对多的部署方式，即一个 T2000 可以同时和多个 NMS 进行对接，一个 NMS 也可以同时管理多个 T2000，只要在部署时，保证被管理 T2000 的名字在 NMS 管理域内唯一即可。通知服务则可以在 T2000 中配置数据发生变更时，及时将详细的变更信息通知 NMS。保证运营商能够随时了解到网络的当前运行状况，并确保 NMS 和 T2000 数据的一致性。

7.3.2 名字服务

名字服务上下文

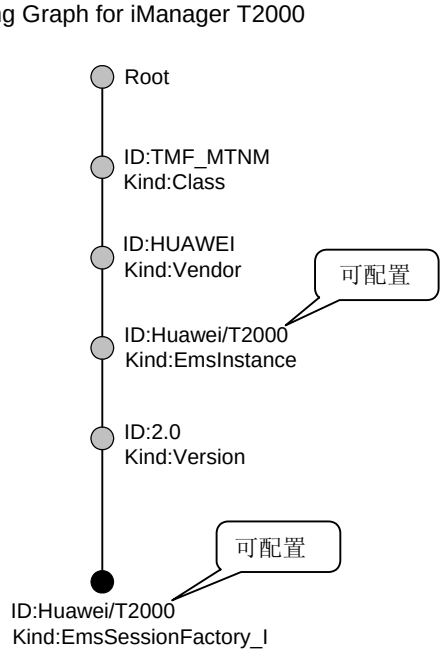
TMF 建议中给出的名字图如图 7-1 所示。

图7-1 TMF 建议的名字图



iManager T2000 按照建议，对 NMS 提供了唯一的访问入口，即 EmsSessionFactory_I 对象的对象引用。iManager T2000 CORBA 接口生成的名字图如图 7-2 所示。

图7-2 iManager T2000 名字图



NMS 可以根据 iManager T2000 CORBA 接口的名字图，借助 CORBA 平台提供的名字服务，完成和 iManager T2000 CORBA 接口的对接。注意其中 T2000 的名字（Huawei/T2000）可以通过系统配置文件进行配置，并可支持多个 iManager T2000 系统同时接入某一 NMS。

说明

T2000 的配置文件$IMAP/corba.cfg 中提供了修改 T2000 名字的选项：EmsName = Huawei/T2000。用户可以根据需要修改该选项的值给 T2000 设置其它的名字，但必须保证该名字在 NMS 管理域内是唯一的。当修改了 T2000 的名字后，名字图中两处标记为"可配置"的值也就随之改变了。

编程示例

下面通过一个简单的示例程序，说明创建上图所示名字图的过程。

步骤 1 定义并实现 NmsSession_I 对象的 Servant 类。

```
class TANmsSession_IImpl : public virtual POA_nmsSession::NmsSession_I
{
public:

    TANmsSession_IImpl();
    ~TANmsSession_IImpl();

    virtual void eventLossOccurred ( const char * startTime,
const char * notificationId    )
                ACE_THROW_SPEC (( CORBA::SystemException));
    virtual void eventLossCleared (    const char * endTime )
                ACE_THROW_SPEC ((CORBA::SystemException));
    virtual session::Session_I_ptr associatedSession ()
```

```
                    ACE_THROW_SPEC (( CORBA::SystemException));
    virtual void ping ( )
                     ACE_THROW_SPEC ((CORBA::SystemException));
    virtual void endSession ( )
                     ACE_THROW_SPEC ((CORBA::SystemException));
};
```

步骤 2 初始化 ORB（object request broker）连接。

```
    CORBA::ORB_var orb = CORBA::ORB_init( argc, argv ) ;
    CORBA::Object_var obj_poa = orb->resolve_initial_references("RootPOA");
    PortableServer::POA_var root_poa = PortableServer::POA::_narrow(obj_poa);
```

步骤 3 获得名字服务的引用。

```
CORBA::Object_var NamingObj = orb->resolve_initial_references( "NameService" );
CosNaming::NamingContext_var NamingContext =
 CosNaming::NamingContext::_narrow( NamingObj.in() );
```

步骤 4 获得 EmsSessionFactory 的引用。

```
    CosNaming::Name name;
    name.length(5);
    name[0].id = CORBA::string_dup("TMF_MTNM");
    name[0].kind = CORBA::string_dup("Class");
    name[1].id = CORBA::string_dup("HUAWEI");
    name[1].kind = CORBA::string_dup("Vendor");
    name[2].id = CORBA::string_dup("Huawei/T2000");
    name[2].kind = CORBA::string_dup("EMSInstance");
    name[3].id = CORBA::string_dup("2.0");
    name[3].kind = CORBA::string_dup("Version");
    name[4].id = CORBA::string_dup("Huawei/T2000");
    name[4].kind = CORBA::string_dup("EmsSessionFactory_I");
    CORBA::Object_var obj;
    try
    {
        obj = NamingContext->resolve( name );
    }
    catch( CORBA::Exception& ex )
    {
        //异常处理;
    }
```

步骤 5 获取 EmsSession 的对象引用并登陆。

```
emsSessionFactory::EmsSessionFactory_I_var EmsSessionFactory =
emsSessionFactory::EmsSessionFactory_I::_narrow( obj.in() );
    TANmsSession_IImpl* pNmsSessionServant = new TANmsSession_IImpl;
    nmsSession::NmsSession_I_var NmsSession = pNmsSessionServant->_this();
    emsSession::EmsSession_I_var emsSessionInterface;
    try{
EmsSessionFactory->getEmsSession( "userName", "userPassword", NmsSession,
emsSessionInterface.out() );
    }catch( CORBA::Exception& ex )
```

```
{
    //异常处理
}
```

步骤 6 获取管理器对象。

```
emsSession::EmsSession_I::managerNames_T_var supportedMgrNameList = NULL;
emsSessionInterface->getSupportedManagers( supportedMgrNameList );
common::Common_I_ptr  pMgr =
    emsSessionInterface->getManager("ManagedElement");
managedElement::ManagedElement_I_var meMgr =
    managedElement::ManagedElement_I::_narrow( pMgr );

//调用 managedElement::ManagedElement_I 对象的方法进行请求交互
meMgr->getAllManagedElements( ….);
```

步骤 7 运行 ORB，启动事件侦听循环。

```
CORBA::Object_var obj_poa = orb->resolve_initial_references("RootPOA");
    PortableServer::POA _var root_poa = PortableServer::POA::_narrow(obj_poa);
root_poa_manager = NMS_POA->the_POAManager();
    root_poa_manager->activate();
orb->run();
```

----结束

7.3.3 通知服务

通知服务支持情况

通知服务的支持情况如表 7-77 所示。

表7-77 通知服务支持表

Notification Service Characteristic	Support 情况	Comment
Push or Pull	Push	不支持 Pull 模式
Number of Notifications Channel	1	每个 EMS 只支持一个 Event Channel
Instance of Notification Service	1	建议一个 EMS 使用一个 Notification Service 实例
EventReliablility	N	
ConnectionReliability	Y	
Usage of Event Priority	N	所有的事件优先级相同
DisCard Policy	FIFO	先进先出
Delivery Order Policy	FIFO	

Notification Service Characteristic	Support 情况	Comment
Start Time Supported	N	不支持
Maximum Batch Size	Y	需要根据实际情况设定该值
Pacing Interval	0	
RejectNewEvents	Y	TRUE
TimeOut	N	

编程示例

下面的一个程序示例，给出了 NMS 如何与 T2000 的通知服务建立连接的过程。

步骤 1 定义并实现 StructuredPushConsumer 对象的 Servant 类。

```
    class TAConsumerImpl : public virtual POA_CosNotifyComm::StructuredPushConsumer
{
public:
    TAConsumerImpl();
    virtual ~TAConsumerImpl();
public:
    virtual void push_structured_event (
const CosNotification::StructuredEvent & notification )
         ACE_THROW_SPEC (( CORBA::SystemException,
         CosEventComm::Disconnected ));
    virtual void disconnect_structured_push_consumer ()
ACE_THROW_SPEC (( CORBA::SystemException ));
    virtual void offer_change (
       const CosNotification::EventTypeSeq & added,
            const CosNotification::EventTypeSeq & removed  )
    ACE_THROW_SPEC (( CORBA::SystemException,
         CosNotifyComm::InvalidEventType));
};
```

步骤 2 获取 EmsSession 的对象引用并登陆。

```
    //具体过程请参考名字服务编程样例
emsSessionFactory::EmsSessionFactory_I_var EmsSessionFactory =
emsSessionFactory::EmsSessionFactory_I::_narrow( obj.in() );
    TANmsSession_IImpl* pNmsSessionServant = new TANmsSession_IImpl;
    nmsSession::NmsSession_I_var NmsSession = pNmsSessionServant->_this();
    emsSession::EmsSession_I_var emsSessionInterface;
    try{
EmsSessionFactory->getEmsSession( "userName", "userPassword", NmsSession,
emsSessionInterface );
    }catch( CORBA::Exception& ex )
    {
        //异常处理
    }
```

步骤 3 获取 EventChannel 的引用。

```
CosNotifyChannelAdmin::EventChannel_var eventChannel;
try{
    emsSessionInterface->getEventChannel( eventChannel );
}catch ( CORBA::Exception& ex )
{
    //异常处理；
}
```

步骤 4 获取 ConsumerAdmin 的对象引用

```
CosNotifyChannelAdmin::ConsumerAdmin_var ca;
try{
ca = channel->new_for_consumers(CosNotifyChannelAdmin::OR_OP, adminID);
}catch( CORBA::Exception& ex)
{
// 异常处理
}
```

步骤 5 订阅事件

```
CosNotification::EventTypeSeq added(1);
CosNotification::EventTypeSeq removed (0);
added.length (1);
removed.length (0);
added[0].domain_name =  CORBA::string_dup ("*");
added[0].type_name = CORBA::string_dup ("*");
try{
        ca->subscription_change( added, removed );
}catch ( CORBA::Exception& ex)
{
        //异常处理
}
```

步骤 6 连接事件通道

```
CosNotifyChannelAdmin::ClientType cType =
        CosNotifyChannelAdmin::STRUCTURED_EVENT;
CosNotifyChannelAdmin::StructuredProxyPushSupplier_ptr pps;
try{
        CosNotifyChannelAdmin::ProxySupplier_var obj =
            ca->obtain_notification_push_supplier( cType, givenID );
        pps =
CosNotifyChannelAdmin::StructuredProxyPushSupplier::_narrow( obj.in() );
}catch( CORBA::Exception& ex )
{
        //异常处理
}

TAConsumerImpl* pConsumerServant = new TAConsumerImpl;
CosNotifyComm::StructuredPushConsumer_var spc = pConsumerServant ->_this();
pps->connect_structured_push_consumer( spc );
```

步骤 7 运行 ORB，启动事件侦听循环。

```
CORBA::Object_var obj_poa = orb->resolve_initial_references("RootPOA");
    PortableServer::POA _var root_poa = PortableServer::POA::_narrow(obj_poa);
root_poa_manager = NMS_POA->the_POAManager();
    root_poa_manager->activate();
orb->run();
```

----结束

7.4 通知事件格式

在 TMF 814 建议的通知事件用法中给出了 CORBA 接口通知事件结构的用法说明。下面是详细的事件结构定义。

7.4.1 NT_ALARM 事件格式

表7-78 NT_ALARM 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证。
objectName	globaldefs::NamingAttributes_T	上报事件的对象名称
nativeEMSName	string	产生告警的对象的相关信息
nativeProbableCause	string	EMS 用户界面上描述的告警原因
objectType	notifications::ObjectType_T	上报该告警的对象的类型
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间），如果网元没有上报该时间值，则为空
isClearable	boolean	标识是否已被清除。
layerRate	transmissionParameters::LayerRate_T	与告警相关的层速率
probableCause	string	TMF 定义的告警可能原因
probableCauseQualifier	string	与 objectName、layerRate 和 probableCause 字段一起来唯一标识一条告警事件类型
perceivedSeverity	notifications::PerceivedSeverity_T	告警级别，对于清除告警，其告警级别为 PS_CLEARED

名称	类型	描述
serviceAffecting	notifications::ServiceAffecting_T	标识该告警是否对业务造成影响 （SA_UNKNOWN / SA_SERVICE_AFFECTING / SA_NON_SERVICE_AFFECTING）
affectedTPList	globaldefs::NamingAttributesList_T	受影响的终结点列表
additionalText	string	产生该告警的对象的详细信息
additionalInfo	globaldefs::NVSList_T	包含告警序列号（AlarmSerialNo）、告警简要原因（AlarmReason）、设备类型（ProductName）、单板名称(EquipmentName)、告警确认状态(AffirmState)、告警参数信息(DetailInfo)六个附加字段
X.733::EventType	string	根据 ITU-T X.733 将告警分为 6 个基本类型： "communicationsAlarm" "qualityofServiceAlarm" "equipmentAlarm" "processingErrorAlarm" "securityAlarm" "environmentalAlarm" 该字段为可选字段。
objectTypeQualifier	string	上报事件的类型，该字段只指针对 TMF3.0 版本中新增的对象类型

说明

T2000 CORBA 接口 NT_ALARM 事件结构中的"nativeEMSName"和"additionalText"字段都是用作存放产生该告警的对象的信息；在"nativeEMSName"字段中产生告警的网元是用 T2000 前台的网元名称表示的，而"additionalText"字段中产生告警的网元是用网元 ID 来表示的；
"additionalInfo"字段用作存放告警序列号，告警简要原因和产生该告警的设备类型。

7.4.2 NT_TCA 事件格式

表7-79 NT_TCA 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证。
objectName	globaldefs::NamingAttributes_T	上报事件的对象名称
nativeEMSName	string	产生性能越限事件的 TP 信息
objectType	notifications::ObjectType_T	产生性能越限告警事件的对象的类型

名称	类型	描述
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间），如果网元没有上报该时间值，则为空。
isClearable	boolean	标识是否已被清除
perceivedSeverity	notifications::PerceivedSeverity_T	性能越限告警事件的级别，统一填写为 PS_INDETERMINATE
layerRate	transmissionParameters::LayerRate_T	与该性能事件相关的层速率
granularity	Granularity_T	性能监视周期值，对普通的性能事件，其周期为 15min 和 24h，对 RMON 性能事件，其周期为 30s，30min 和可设周期。
pmParameterName	PMParameterName_T	性能参数名称
pmLocation	PMLocation_T	标识性能参数测量点所在 TP 的位置 （PML_NEAR_END_Rx / PML_FAR_END_Rx / PML_NEAR_END_Tx / PML_FAR_END_Tx / PML_BIDIRECTIONAL）
thresholdType	PMThresholdType_T	性能越限类型 （TWM_HIGHEST / TWM_HIGH / TWM_LOW / TWM_LOWEST）
value	float	产生性能越限时的性能参数值
unit	string	性能事件的单位
additionalInfo	globaldefs::NVSList_T	TCA 事件附加字段，目前包括产生 TCA 的事件的设备类型信息（ProductName）和 单板名称信息（EquipmentName）。

说明

T2000 CORBA 接口 NT_TCA 事件结构中的 "additionalInfo"字段为扩展字段，用作存放产生该性能越限事件的设备类型。

7.4.3 NT_ FILE_TRANSFER_STATUS 事件格式

表7-80 NT_ FILE_TRANSFER_STATUS 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
fileName	string	文件的名字
transferStatus	FileTransferStatus_T	文件传输状态 （FT_IN_PROGRESS / FT_FAILED / FT_COMPLETED）
percentComplete	short	文件传输完成度，取值为 0-100
failureReason	string	文件传输失败时失败原因的描述

7.4.4 NT_OBJECT_CREATION 事件格式

表7-81 NT_OBJECT_CREATION 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
objectName	globaldefs::NamingAttributes_T	创建的对象的名称
objectType	notifications::ObjectType_T	创建的对象的类型
objectTypeQualifier	string	上报事件的类型，该字段只指针对 TMF3.0 版本中新增的对象类型
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间），如果网元没有上报该时间值，则为空
edgePointRelated	boolean	如果事件是相关于边界点的 PTP 或者相关于边界点 PTP 的保护组，则该值为真；否则为假。 这个字段是可选的。

说明

NT_OBJECT_CREATION 事件结构还包含了一个很重要的部分：remainder_of_body，它描述了被创建对象的详细信息。但是 remainder_of_body 的具体结构并不是固定的，而是根据创建对象类型不同而不同的。下面以"创建网元"事件结构和"创建单板"事件结构为例说明：

表7-82 NT_OBJECT_CREATION 事件结构补充（remainder_of_body）_创建网元

名称	类型	描述
name	globaldefs::NamingAttributes_T	网元名称
userLabel	string	用户标签
nativeEMSName	string	创建的对象在 EMS 界面上的名称
owner	string	对象所有者
location	string	网元所在位置
version	string	网元主控板的版本号
productName	string	网元的产品名称
communicationState	CommunicationState_T	网元通讯状态（CS_AVAILABLE / CS_UNAVAILABLE）
emsInSyncState	boolean	标识是否与 EMS 同步
supportedRates	transmissionParameters::LayerRateList_T	网元支持的层速率
additionalInfo	globaldefs::NVSList_T	附加信息（存放地域、网关、告警级别等简要信息）

表7-83 NT_OBJECT_CREATION 事件结构补充（remainder_of_body）_创建单板

名称	类型	描述
name	globaldefs::NamingAttributes_T	单板名称
userLabel	string	用户标签
nativeEMSName	string	创建的对象在 EMS 界面上的名称
owner	string	对象所有者
alarmReportingIndicator	boolean	告警自动上报开关

名称	类型	描述
serviceState	ServiceState_T	单板当前状态 （IN_SERVICE / OUT_OF_SERVICE / OUT_OF_SERVICE_BY_MAINTENANCE / SERV_NA）
expectedEquipmentObjectType	EquipmentObjectType_T	逻辑单板类型
installedEquipmentObjectType	EquipmentObjectType_T	物理单板类型
installedPartNumber	string	物理单板的资源编号
installedVersion	string	物理单板的版本号
installedSerialNumber	string	物理单板的序列号
additionalInfo	globaldefs::NVSList_T	创建的单板的一些附加信息

7.4.5 NT_OBJECT_DELETION 事件格式

表7-84 NT_OBJECT_DELETION 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
objectName	globaldefs::NamingAttributes_T	被删除的对象的标示符
objectType	notifications::ObjectType_T	被删除对象类型的标示符。这个参数仅被用于 2.1 或更早的版本中。
objectTypeQualifier	string	上报事件的类型，该字段只指针对 TMF3.0 版本中新增的对象类型
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间）。网元不能上报时间的情况下，这个值是可选的，或者可上报为空串。

名称	类型	描述
edgePointRelated	boolean	如果事件是相关于边界点的 PTP 或者相关于边界点 PTP 的保护组，则该值为真；否则为假。 这个字段是可选的。

7.4.6 NT_ATTRIBUTE_VALUE_CHANGE 事件格式

表7-85 NT_ATTRIBUTE_VALUE_CHANGE 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
objectName	globaldefs::NamingAttributes_T	发生属性改变的对象的名称
objectType	notifications::ObjectType_T	发生属性改变的对象的类型。这个参数仅被用于 2.1 或更早的版本中。
objectTypeQualifier	string	上报事件的类型，该字段只指针对 TMF3.0 版本中新增的对象类型
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间）。网元不能上报时间的情况下，这个值是可选的，或者可上报为空串。
edgePointRelated	boolean	如果事件是相关于边界点的 PTP 或者相关于边界点 PTP 的保护组，则该值为真；否则为假。 这个字段是可选的。
attributeList	notifications::NVList_T	发生改变的属性值列表。

7.4.7 NT_STATE_CHANGE 事件格式

表7-86 NT_STATE_CHANGE 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
objectName	globaldefs::NamingAttributes_T	发生状态改变的对象名称
objectType	notifications::ObjectType_T	发生状态改变的对象的类型。这个参数仅被用于 2.1 或更早的版本中。
objectTypeQualifier	string	上报事件的类型，该字段只指针对 TMF3.0 版本中新增的对象类型
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间）。网元不能上报时间的情况下，这个值是可选的，或者可上报为以空串。
edgePointRelated	boolean	如果事件是相关于边界点的 PTP 或者相关于边界点 PTP 的保护组，则该值为真；否则为假。 这个字段是可选的。
attributeList	notifications::NVList_T	发生改变的状态值列表。

7.4.8 NT_PROTECTION_SWITCH 事件格式

表7-87 NT_PROTECTION_SWITCH 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间）。网元不能上报时间的情况下，这个值是可选的，或者可上报为空串。

名称	类型	描述
ProtectionType	protection::ProtectionType_T	保护的类型。
switchReason	protection::switchReason_T	倒换发生的原因。
layerRate	transmissionParameters::LayerRate_T	与倒换相关联的层速率。
groupName	globaldefs::NamingAttributes_T	发生倒换的保护组的标示符。如果保护类型为SNCP，这个字段可选或者可以上报为空。
protectedTP	globaldefs::NamingAttributes_T	倒换发生时被保护 TP 的标示符。如果是SNCP，这个值一般是固定的。如果是一个 2 纤 BLSR 环倒换通知，这个值是倒换期间未激活的 TP 的表示符。如果是一个 4 纤 MSP 环倒换通知，这是一个在倒换期间未激活的工作 TP。如果是一个 1:N MSP 倒换通知，这个是发生保护倒换的工作 TP。如果是反向 1+1 MSP 倒换通知，这个值一般为工作 TP；但如果是非反向的 1+1 MSP 通知，这个值是倒换前处于激活状态的 TP（倒换后被保护的 TP 发生了改变）。
switchAwayFromTP	globaldefs::NamingAttributes_T	发生倒换时的源 TP。如果是一个 2 纤 MSP 环倒换，就是发生倒换的那个 TP。对于一个 4 纤 MSP 环倒换，这个值是 MSP 1:N 保护组（工作或保护）内的一个 TP。对于 4 纤环，是发生倒换的工作 TP，例如从东向倒换到西向，这个值就是一个东向工作 TP。
switchToTP	globaldefs::NamingAttributes_T	发生倒换时的目的 TP。这个 TP 是倒换后被激活的源 TP，或者当前处于激活状态的 TP。如果是 2 纤 BLSR，当倒换状态返回到正常状态时，这个值就是已被倒换的 TP。如果是 4 纤 BLSR 环倒换，当倒换状态返回到正常状态时，这个值就是发生倒换的工作 TP，例如从东向倒换到西向时，这个值就是东向工作 TP。
nativeEMSName	string	发生保护倒换的保护组名称
additionalInfo	notifications::NVList_T	保护倒换事件的附加字段，目前包含了发生倒换设备的设备类型信息（ProductName）、保护单板信息（protectedEquipmentName）、倒换源单板信息（switchAwayFromEquipmentName）、倒换宿单板信息（switchToEquipmentName）。

说明

T2000 CORBA 接口 NT_PROTECTION_SWITCH 事件结构中的"additionalInfo"字段为扩展字段，用作存放发生保护倒换的设备类型。

7.4.9 NT_ATMPROTECTION_SWITCH 事件格式

表7-88 NT_ATMPROTECTION_SWITCH 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间），如果网元没有上报该时间值，则为空
pgName	globaldefs::NamingAttributes_T	发生倒换的 ATM 保护组名称
protectType	HW_mstpProtection::HW_AtmProtectType_T	保护类型
srcEndSwitchPara	HW_mstpProtection::HW_AtmPGSingleEndSwitchPara_T	源端倒换参数，描述了源端的倒换原因，倒换状态及其其它一些倒换信息
snkEndSwitchPara	HW_mstpProtection::HW_AtmPGSingleEndSwitchPara_T	宿端倒换参数，描述了宿端的倒换原因，倒换状态及其其它一些倒换信息

7.4.10 NT_WDMPROTECTION_SWITCH 事件格式

表7-89 NT_WDMPROTECTION_SWITCH 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间）。网元不能上报时间的情况下，这个值是可选的，或者可上报为空串。
protectionGroupType	protection::WDMProtectionGroup_T	保护组类型。
switchReason	protection::switchReason_T	倒换发生的原因。
groupName	globaldefs::NamingAttributes_T	发生倒换的保护组名称。

名称	类型	描述
protectedTP	globaldefs::NamingAttributes_T	倒换发生时被保护 TP 名称。
switchAwayFromTP	globaldefs::NamingAttributes_T	发生倒换的源 TP 名称。
switchToTP	globaldefs::NamingAttributes_T	发生倒换的目的 TP 名称。
nativeEMSName	string	发生倒换的波分保护组信息
additionalInfo	notifications::NVList_T	保护倒换事件的附加字段，给出了发生倒换设备的设备类型信息（ProductName）、保护单板信息（protectedEquipmentName）、倒换源单板信息（switchAwayFromEquipmentName）、倒换宿单板信息（switchToEquipmentName）。

说明

T2000 CORBA 接口 NT_WDMPROTECTION_SWITCH 事件结构中的"additionalInfo"字段为扩展字段，用作存放发生波分保护倒换的设备类型。

7.4.11 NT_RPRPROTECTION_SWITCH 事件格式

表7-90 NT_RPRPROTECTION_SWITCH 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
objectName	globaldefs::NamingAttributes_T	发生倒换的 RPR 节点名称
objectType	notifications::ObjectType_T	发生倒换的对象的类型
objectTypeQualifier	string	上报事件的类型，该字段只指针对 TMF3.0 版本中新增的对象类型
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间）。网元不能上报时间的情况下，这个值是可选的，或者可上报为空串。
edgePointRelated	boolean	如果事件是相关于边界点的 PTP 或者相关于边界点 PTP 的保护组，则该值为真；否则为假。 这个字段是可选的。
attributeList	notifications::NVList_T	指出了发生 RPR 倒换的保护组详细信息，包括倒换状态，原因，东西向的倒换参数等。

7.4.12 NT_EPROTECTION_SWITCH 事件格式

表7-91 NT_EPROTECTION_SWITCH 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间），如果网元没有上报该时间值，则为空
eProtectionGroupType	protection::EProtectionGroupType_T	上报的设备保护类型
eSwitchReason	protection::ESwitchReason_T	倒换发生的原因
groupName	globaldefs::NamingAttributes_T	发生倒换的设备组名称
protectedE	globaldefs::NamingAttributes_T	指明倒换发生时被保护的设备。对 M：N 类型的保护组，被保护设备指向的是倒换发生时的工作设备
switchAwayFromE	globaldefs::NamingAttributes_T	倒换发生的源设备
switchToE	globaldefs::NamingAttributes_T	倒换发生的目的设备
nativeEMSName	string	发生倒换的设备保护组详细信息
additionalInfo	globaldefs::NVSList_T	附加字段，目前包括发生设备保护倒换的设备类型信息（ProductName）、保护单板信息（protectedEquipmentName）、倒换源单板信息（switchAwayFromEquipmentName）、倒换宿单板信息（switchToEquipmentName）。

说明

T2000 CORBA 接口 NT_EPROTECTION_SWITCH 事件结构中的"additionalInfo"字段为扩展字段，用作存放发生设备保护倒换的设备类型信息。

7.4.13 NT_ROUTE_CHANGE 事件格式

表7-92 NT_ROUTE_CHANGE 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
objectName	globaldefs::NamingAttributes_T	上报事件的对象名称
objectType	notifications::ObjectType_T	上报事件的对象类型
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间）。网元不能上报时间的情况下，这个值是可选的，或者可上报为空串。
routeChangeEvent	subnetworkConnection::RerouteChangeEvent_T	路由改变过程中的阶段状态信息 （RerouteStarted / RerouteCompleted / RerouteFailed）
route	subnetworkConnection::Route_T	新路由详细信息

7.4.14 NT_ASON_RESOURCE_CHANGE 事件格式

表7-93 NT_ASON_RESOURCE_CHANGE 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
objectName	globaldefs::NamingAttributes_T	上报事件的对象名称
objectTypeQualifier	string	指明上报事件的主体对象，是智能域还是智能节点 （ROUTING_AREA / ROUTING_NODE）
notifyType	string	指明上报事件的类型，是创建还是删除（OBJECT_CREATE / OBJECT_DELETE）
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）

名称	类型	描述
neTime	globaldefs::Time_T	网元上报事件的时间值（为 UTC 时间），如果网元没有上报该时间值，则为空
edgePointRelated	boolean	如果事件是相关于边界点的 PTP 或者相关于边界点 PTP 的保护组，则该值为真；否则为假。 这个字段是可选的。

7.4.15 NT_PRBSTEST_STATUS 事件格式

表7-94 NT_PRBSTEST_STATUS 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
TPName	globaldefs::NamingAttributes_T	进行 2M 伪随机码测试的 TP 名称
TestStatus	notifications::FileTransferStatus_T	2M 伪随机码测试状态 （FT_COMPLETED / FT_IN_PROGRESS）
percentComplete	short	测试进度，取值为 0-100

7.4.16 NT_HEARTBEAT 事件格式

表7-95 NT_HEARTBEAT 事件格式

名称	类型	描述
notificationId	string	通知事件的唯一 ID 编号。除非支持 X.733::CorrelatedNotifications 参数，否则该 ID 的顺序性不能保证
objectName	globaldefs::NamingAttributes_T	上报事件的对象名称
objectType	notifications::ObjectType_T	上报事件的对象类型
emsTime	globaldefs::Time_T	EMS 上报事件的时间值（为 UTC 时间）

7.5 层速率说明

表7-96 T2000 CORBA 接口支持的层速率列表

序号	层速率枚举描述	说明
1	LR_Not_Applicable	the layer is not relevant
2	LR_T1_and_DS1_1_5M	1.5 Mbit/s async/PDH signal
3	LR_T2_and_DS2_6M	6 Mbit/s async/PDH signal
4	LR_T3_and_DS3_45M	45 Mbit/s async/PDH signal
5	LR_E1_2M	2Mbit/s PDH signal
6	LR_E2_8M	8Mbit/s PDH signal
7	LR_E3_34M	34 Mbit/s PDH signal
8	LR_E4_140M	140 Mbit/s PDH signal
9	LR_E5_565M	565 Mbit/s PDH signal
10	LR_VT1_5_and_TU11_VC11	VC11 SONET/SDH path signal
11	LR_VT2_and_TU12_VC12	VC12 SONET/SDH path signal
12	LR_VT6_and_TU2_VC2	VC2 SONET/SDH path signal
13	LR_Low_Order_TU3_VC3	VC3 SONET/SDH path signal
14	LR_STS1_and_AU3_High_Order_VC3	AU3 SONET/SDH path signal
15	LR_STS3c_and_AU4_VC4	SONET/SDH path signal
16	LR_STS12c_and_VC4_4c	12xSTS-1/4xVC4 Contiguous Concatenation
17	LR_STS48c_and_VC4_16c	48xSTS-1/16xVC4 Contiguous Concatenation
18	LR_STS192c_and_VC4_64c	192xSTS-1/64xVC4 Contiguous Concatenation
19	LR_Section_OC1_STS1_and_RS_STM0	STM-0 regenerator section
20	LR_Section_OC3_STS3_and_RS_STM1	STM-1 regenerator section
21	LR_Section_OC12_STS12_and_RS_STM4	STM-4 regenerator section
22	LR_Section_OC48_STS48_and_RS_STM16	STM-16 regenerator section
23	LR_Section_OC192_STS192_and_RS_STM64	STM-64 regenerator section
24	LR_Line_OC1_STS1_and_MS_STM0	STM-0 multiplex section
25	LR_Line_OC3_STS3_and_MS_STM1	STM-1 multiplex section
26	LR_Line_OC12_STS12_and_MS_STM4	STM-4 multiplex section

序号	层速率枚举描述	说明
27	LR_Line_OC48_STS48_and_MS_STM16	STM-16 multiplex section
28	LR_Line_OC192_STS192_and_MS_STM64	STM-64 multiplex section
40	LR_Optical_Channel	for WDM wavelength
41	LR_Optical_Multiplex_Section	for WDM wavelength bands
42	LR_Optical_Transmission_Section	for WDM entire optical signal, i.e. used for OTS and OMS layers of OTM-n.m (n≥1)
43	LR_ATM_NI	for ATM Network Interfaces (UNI and NNI)
44	LR_ATM_VP	for ATM Virtual Paths
45	LR_ATM_VC	for ATM Virtual Channels
46	LR_PHYSICAL_ELECTRICAL	analogue signal on electrical physical media
47	LR_PHYSICAL_OPTICAL	analogue signal on optical physical media
49	LR_OPTICAL_SECTION	represents the wavelength termination for a non DWDM system, i.e. used for all kinds of single-lambda ports
50	LR_DIGITAL_SIGNAL_RATE	raw binary electrical signal of unspecified rate
58	LR_D1_Video	video capable port
59	LR_ESCON	IBM protocol for mainframes
61	LR_Fast_Ethernet	TMF 不支持
62	LR_FC_12_133M	133 Mbit/s Fibre Channel protocol
63	LR_FC_25_266M	266 Mbit/s Fibre Channel protocol
64	LR_FC_50_531M	531 Mbit/s Fibre Channel protocol
65	LR_FC_100_1063M	1063 Mbit/s Fibre Channel protocol
66	LR_FDDI	
67	LR_FICON	IBM Protocol for mainframes
68	LR_Gigabit_Ethernet	TMF 不支持
72	LR_DSR_OC1_STM0	STM-0 digital signal rate
73	LR_DSR_OC3_STM1	STM-1 digital signal rate
74	LR_DSR_OC12_STM4	STM-4 digital signal rate
75	LR_DSR_OC24_STM8	STM-8 digital signal rate

序号	层速率枚举描述	说明
76	LR_DSR_OC48_and_STM16	STM-16 digital signal rate
77	LR_DSR_OC192_and_STM64	STM-64 digital signal rate
78	LR_DSR_OC768_and_STM256	STM-256 digital signal rate
79	LR_DSR_1_5M	1,5 Mbit/s digital signal rate
80	LR_DSR_2M	2 Mbit/s digital signal rate
81	LR_DSR_6M	4 Mbit/s digital signal rate
82	LR_DSR_8M	8 Mbit/s digital signal rate
83	LR_DSR_34M	34 Mbit/s digital signal rate
84	LR_DSR_45M	45 Mbit/s digital signal rate
85	LR_DSR_140M	140 Mbit/s digital signal rate
86	LR_DSR_565M	565 Mbit/s digital signal rate
87	LR_DSR_Gigabit_Ethernet	Gigabit_Ethernet digital signal rate
88	LR_Section_OC24_STS24_and_RS_STM8	STM-8 regenerator section
89	LR_Line_OC24_STS24_and_MS_STM8	STM-8 multiplex section
90	LR_Section_OC768_STS768_and_RS_STM256	STM-256 regenerator section
91	LR_Line_OC768_STS768_and_MS_STM256	STM-256 multiplex section
93	LR_DSR_2xSTM1	2 times STM-1 radio multiplexing
96	LR_Ethernet	all Ethernet rates
97	LR_DSR_Fast_Ethernet	10/100 Mbit/s Ethernet
98	LR_Encapsulation	for Ethernet, the following encapsulation protocols apply: HDLC/PPP, HDLC/LAPS, ML/PPP, and GFP Transparent or Frame Mapped types
99	LR_Fragment	used for inverse multiplexing modeling (Virtual Concatenation for SONET/SDH and IMA)
100	LR_STS6c_and_VC4_2c	6xSTS-1/2xVC4 Contiguous Concatenation
101	LR_STS9c_and_VC4_3c	9xSTS-1/3xVC4 Contiguous Concatenation
29	LR_STS24c_and_VC4_8c	HUAWEI 扩充
113	LR_DSR_10Gigabit_Ethernet	10 Gbit/s Ethernet
8001	LR_Section_and_RS	HUAWEI 扩充

序号	层速率枚举描述	说明
8002	LR_Line_and_MS	HUAWEI 扩充
8003	LR_ATM	ATM 层速率[HUAWEI 扩充]
8004	LR_Optical_Supervision_Channel	光监控层速率[HUAWEI 扩充]
8005	LR_FC_200_2125M	2125 Mbit/s Fibre Channel protocol[HUAWEI 扩充]

7.6 传输参数说明

T2000 CORBA 接口支持的传输参数如表 7-97 所示：

表7-97 传输参数说明

Parameter Name	Layers	Legalvalues	Potentially settable from	Comment
Generally applicable parameters				
TrailTraceExpectedRx	All PDH,SDH,SONET layers where Trail Trace is specified for, and LR_PHYSICAL_MEDIALESS for radio	a string of up to 64 chars	EMS & NMS	应收 J2[支路板 PTP]、J2[VC12 CTP] 应收 J1[VC12 CTP] 应收 J0,J1
TrailTraceActualTx	All PDH,SDH,SONET layers where Trail Trace is specified for, and LR_PHYSICAL_MEDIALESS for radio	a string of up to 64 chars	EMS & NMS	实发 J1、J2[支路板 PTP] 实发 J1、J2[VC12 CTP] 应发 J1，实发 J0
TrailTraceActualRx	All PDH,SDH,SONET layers where Trail Trace is specified for, and LR_PHYSICAL_MEDIALESS for radio	a string of up to 64 chars	EMS	实收 J1、J2[支路板 PTP] 实收 J1、J2[VC12 CTP] 实收 J0、J1

Parameter Name	Layers	Legalvalues	Potentially settable from	Comment
ClientType	DSR layers and OCH	"SONET", "SDH",	EMS & NMS	客户侧业务类型
SDH/SONET/PDH specific parameters				
SignalLabelActualTx	All SDH,SONET path layers	This is in decimal format with values between 0 and 255	EMS	应发 C2
SignalLabelExpectedRx	All SDH/SONET path layers	This is in decimal format with values between 0 and 255	EMS & NMS	应收 C2
ATM specific parameters (ATM NI CTP, ATM VP CTP, ATM VC CTP, ATM NI TPPool, ATM VP TPPool)				
ATMNIType	LR_ATM_N (CTP) I	"UNI","NNI"	EMS & NMS	ATM 端口[以下简称 ATM]和 ATM 内部端口[以下简称 ATMTrunk]的端口类型
MaxVPIBits	LR_ATM_NI (CTP)	Number [0,12]	EMS & NMS	最大 VPI 位数 [ATM/ATMTrunk]
MaxVCIBits	LR_ATM_NI (CTP)	Number [0,16]	EMS & NMS	最大 VCI 位数 [ATM/ATMTrunk]
MaxVPC	LR_ATM_NI (CTP, TPPool)	A number n.	EMS & NMS	最大 VPC 位数 [ATM/ATMTrunk]
MaxVCC	LR_ATM_NI (CTP, TPPool)	A number n	EMS & NMS	最大 VCC 位数 [ATM/ATMTrunk]
ConfiguredVPC	LR_ATM_NI	Number	EMS & NMS	配置的 VPC 个数 [ATM/ATMTrunk]
ConfiguredVCC	LR_ATM_NI	Number	EMS & NMS	配置的 VCC 个数 [ATM/ATMTrunk]
ActivatedStatus	LR_ATM_NI	"Activated", "Unactivated"	EMS & NMS	激活状态 [ATM/ATMTrunk]
UPC/NPC	LR_ATM_NI	"Enable","Disable"	EMS & NMS	UPC/NPC[ATM/ATMTrunk]
LoopBack	LR_ATM_NI	"NoLoopBack", "InLoopBack", "OutLoopBack"	EMS & NMS	端口环回方式 [ATM/ATMTrunk]

Parameter Name	Layers	Legalvalues	Potentially settable from	Comment
WDM specific parameters				
MaxNumberOCh	OMS	Integer	EMS	最大波长数
FrequencySpacing	OMS	This is in a decimal format and in terms of Ghz.	EMS	波长频率间隔
ClientRate	DSR layers and OCH	"nnnn.ppp" in megabits per second Decimal notation in a string format.	EMS	客户侧当前速率
MaxClientRate				客户侧最大速率
MinClientRate	DSR layers and OCH	"nnn.pp"The unit of these rates are Mbps. Decimal notation in a string format.	EMS	客户侧最小速率
AutoNegotiation	LR_PHYSICAL_OPTIX	"disable" "enable" "NA"	EMS	自协商使能状态
LaserStatus	LR_PHYSICAL_OPTIX	"off" "on" "NA"	EMS	激光器使能状态
Protection specific parameters				
SPRINGNodeId	All SDH/SONET connectable layers	0.."15" (may be any other string that matches the SPRINGNodeId attribute of PGs), "Unknown"	EMS & NMS	本地环节点号 [MSP 保护环(0-15), 其它的一律为 "Unknown"]
ProtectionRole	All connectable layers	"Primary","Backup"	EMS	VC4 的保护角色 [SNCP: 工作/保护]
TCM related parameters for SDH/SONET				
TCMTrailTraceActualTx	SDH/SONET path layer rates	a string of up to 64 chars	EMS & NMS	应发串接监视踪迹字节
TCMContraTrailTraceExpectedRx	SDH/SONET path layer rates	a string of up to 64 chars	EMS & NMS	应收串接监视踪迹字节
TCMContraTrailTraceActualRx	SDH/SONET path layer rates	a string of up to 64 chars	EMS	实收串接监视踪迹字节

Parameter Name	Layers	Legalvalues	Potentially settable from	Comment
Ethernet specific parameters				
Tag	LR_Ethernet	"Tag Aware", "Access", "Hybrid"	EMS & NMS	以太网 MAC 端口[以下简称 MAC]和以太网 MP 端口[以下简称 MP]Tag 标识
VlanID	LR_Ethernet	Number	EMS & NMS	VlanID[MAC/MP]
UserID	LR_Ethernet	Number	EMS & NMS	UserID[MAC/MP]
VLanPriority	LR_Ethernet	Number[0…7]	EMS & NMS	Vlan 优先级[MAC/MP]
PortEnable	LR_Ethernet	"Enable","Disable"	EMS & NMS	端口使能[MAC]
FlowControlEnable	LR_Ethernet	"Enable","Disable"	EMS & NMS	流控使能[MAC]
WorkingMode	LR_Ethernet	"Auto","10MHalfDuplex","10MFullDuplex","100MHalfDuplex","100MFullDuplex","1000MHalfDuplex","1000MFullDuplex","DptMode"	EMS & NMS	工作模式[MAC]
MaxPacketLength	LR_Ethernet	Number[1512-65535]	EMS & NMS	最大包长[MAC]
PhysicalParameters	LR_Ethernet	Number	EMS & NMS	物理参数[MAC]
PortType	LR_Ethernet	"PE","P"	EMS & NMS	端口类型[MAC/MP]
EncapsulateFormat	LR_Ethernet	"MartinioE","CCCoE","VMANoE","MartinioP"	EMS & NMS	封装格式[MAC/MP]
LaserSwitch	LR_Ethernet	"true","false"	EMS & NMS	激光器[MAC]
EnablePPT	LR_Ethernet	"Enable","Disable"	EMS & NMS	PPT 使能[MAC]
DefaultForwardPriority	LR_Ethernet	Number	EMS & NMS	默认转发优先级[MAC]
NonAutoNegotiationFlowControlMode	LR_Ethernet	"Disable","Symmetric","AutoNeg_Non-Symmetric","AutoNeg_Symmetric","AutoNeg_Symmetric/Non-Sysmmetric","ReceiveOnly","SendOnly"	EMS & NMS	非自动流控模式[MAC]

Parameter Name	Layers	Legalvalues	Potentially settable from	Comment
AutoNegotiationFlowControlMode	LR_Ethernet	"Disable","Symmetric","AutoNeg_Non-Symmetric","AutoNeg_Symmetric","AutoNeg_Symmetric/Non-Sysmmetric","ReceiveOnly","SendOnly"	EMS & NMS	自动流控模式[MAC]
ActualFlowControlMode	LR_Ethernet	"Disable","Symmetric","AutoNeg_Non-Symmetric","AutoNeg_Symmetric","AutoNeg_Symmetric/Non-Sysmmetric","ReceiveOnly","SendOnly"	EMS & NMS	实际流控模式[MAC]
BroadcastMsgSuppress	LR_Ethernet	"Enable","Disable"	EMS & NMS	消息广播[MAC]
BroadcastMsgSuppressThreshold	LR_Ethernet	Number	EMS & NMS	广播门限[MAC]
NestedVlanID	LR_Ethernet	Number	EMS & NMS	Nested VlanID [MAC]
NestedPriority	LR_Ethernet	Number	EMS & NMS	Nested 优先级[MAC]
PTTagEnable	LR_Ethernet	"Enable","Disable"	EMS & NMS	PTT 模式[MAC]
ServiceMode	LR_Ethernet	"SW","PKT"	EMS & NMS	业务模式[MAC/MP]
EntranceDetect	LR_Ethernet	"Enable","Disable"	EMS & NMS	入口检测[MAC/MP]
EncapsulateProtocol	LR_Ethernet	"HDLC","GFP","ATM","PPP","LAPS","CRC","CRC_CFP"	EMS & NMS	协议封装格式[MP]
EnableLPT	LR_Ethernet	"Enable","Disable"	EMS & NMS	LPT 是否使能[MP]
Scramble	LR_Ethernet	"Not Scramble","Scrambling Mode[X43+1]","Scrambling Mode[X48+1]"	EMS & NMS	混淆模式[MP]
WithExtendedHeader	LR_Ethernet	"true","false"	EMS & NMS	是否扩展包头[MP]
CheckFieldLength	LR_Ethernet	"FCS32","FCS16","No"	EMS & NMS	校验方式[MP]
CRCReverse	LR_Ethernet	"Reverse","Unreverse"	EMS & NMS	CRC 校验[MP]

Parameter Name	Layers	Legalvalues	Potentially settable from	Comment
FCSCalculateSeq	LR_Ethernet	"Big endian","Little endian"	EMS & NMS	FCS 的计算位序[MP]
SharedAttributes	LR_Ethernet	"yes","no"	EMS & NMS	共享属性[MP]
AutoNegotiation	LR_DSR_Fast_Ethernet	"Enabled", "Disabled"	EMS & NMS	光口自协商

7.7 AdditionalInfo 使用说明

本章介绍了 T2000 CORBA 接口各功能模块中附加字段的使用情况。其中附加字段指 additionalInfo 和 additionalCreationInfo 两部分。包括如下类型：

- ManagedElement_T
- TerminationPoint_T
- EMS_T
- Equipment_T
- EquipmentHolder_T
- SubnetworkConnection_T
- SNCCreateData_T
- SNCModifyData_T
- ProtectionSubnetwork_T
- CrossConnect_T
- HW_EthService_T
- EthernetOAMOperation_T
- ELLinkCreateData_T
- FDFrCreateData_T
- NT_ALARM
- NT_TCA
- NT_PROTECTION_SWITCH
- NT_EPROTECTION_SWITCH
- NT_WDMPROTECTION_SWITCH

7.7.1 ManagedElement_T

字段名称	字段含义说明	字段取值说明	备注
PhyInfo	网元所属的机房信息	FREE STRING	如果是光网元则没有该字段，相应的信息会上报到子架对象（EquipmentHolder）的附加字段中
GateWay	网元所属的网关 IP 地址	FIXED STRING， 形如："590175/127.0.0.1" (第一个字段为所属网关网元 ID，第二个字段为所属网关的 IP 地址)	如果是光网元则没有该字段，相应的信息会上报到子架对象（EquipmentHolder）的附加字段中
PSNName	网元所属的保护子网信息	FREE STRING	如果是光网元则没有该字段，相应的信息会上报到子架对象（EquipmentHolder）的附加字段中
AlarmSeverity	网元当前最高告警级别	FIXED STRING， 取值范围： "PS_CRITICAL", "PS_MAJOR", "PS_MINOR", "PS_WARNING", "PS_INDETERMINATE", "PS_CLEARED"	
ShelfType	网元的子架类型	FIXED STRING， 取值范围： “ST_TYPE1”, "ST_TYPE2", "ST_TYPE3", "ST_STANDARD", "ST_ENHANCED", "ST_EXTENDED", "ST_SS41", "ST_SS42", "ST_R001_SDH", "ST_R001_SONET", "ST_R002", "ST_MetroR001", "ST_MetroR002", "ST_INTEGRATED", "ST_FRONTACCESSED", "ST_WDMSTANDARD",	如果是光网元则没有该字段，相应的信息会上报到子架对象（EquipmentHolder）的附加字段中

字段名称	字段含义说明	字段取值说明	备注
		"ST_WDMEXTENDED", "ST_BOX", "ST_EXBOX", "ST_OSN1500A", "ST_OSN1500B", "ST_OSN3500T", "ST_INVALID"	
PreConfig	网元预配置信息	FIXED STRING， 取值范围： “0”, “1”	如果是光网元则没有该字段，相应的信息会上报到子架对象（EquipmentHolder）的附加字段中

7.7.2 TerminationPoint_T

字段名称	字段含义说明	取值说明	备注
ServiceLoadFlag	业务装载指示	FIXED STRING， 取值范围： "0", "1" 分别表示未装载和装载	应用于支路端口上

7.7.3 EMS_T

字段名称	字段含义说明	取值说明	备注
AlarmSeverity	网管当前最高告警级别	FIXED STRING， 取值范围： "PS_CRITICAL", "PS_MAJOR", "PS_MINOR", "PS_WARNING", "PS_INDETERMINATE", "PS_CLEARED"	
Location	网管所属在物理位置	FREE STRING	

字段名称	字段含义说明	取值说明	备注
CommuAddress	CORBA Agent 的通讯 IP 和端口信息	FIXED STRING， 形如:"127.0.0.1:12003" 127.0.0.1 为 CORBA Agent 通讯 IP，12003 为 CORBA Agent 通讯端口	

7.7.4 Equipment_T

字段名称	字段含义说明	取值说明	备注
AlarmSeverity	设备当前最高告警级别	FIXED STRING， 取值范围： "PS_CRITICAL", "PS_MAJOR", "PS_MINOR", "PS_WARNING", "PS_INDETERMINATE", "PS_CLEARED"	

7.7.5 EquipmentHolder_T

字段名称	字段含义说明	取值说明	备注
AlarmSeverity	WDM 子架当前最高告警级别	FIXED STRING， 取值范围： "PS_CRITICAL", "PS_MAJOR", "PS_MINOR", "PS_WARNING", "PS_INDETERMINATE", "PS_CLEARED"	只有 WDM 子架支持该属性
PhyInfo	WDM 子架所属的机房	FREE STRING	只有 WDM 子架支持该属性
GateWay	WDM 子架所属的网关 IP 地址	FIXED STRING， 形如："590175/127.0.0.1" (第一个字段为所属网关网元 ID，第二个字段为所属网关的 IP 地址)	只有 WDM 子架支持该属性

字段名称	字段含义说明	取值说明	备注
Version	WDM 子架的版本	FIXED STRING，形如: "5.8.3.10"为网元版本	只有 WDM 子架支持该属性
ProductName	WDM 子架类型	FREE STRING	只有 WDM 子架支持该属性
ShelfType	子架类型	FIXED STRING， 取值范围： “ST_TYPE1”, "ST_TYPE2", "ST_TYPE3", "ST_STANDARD", "ST_ENHANCED", "ST_EXTENDED", "ST_SS41", "ST_SS42", "ST_R001_SDH", "ST_R001_SONET", "ST_R002", "ST_MetroR001", "ST_MetroR002", "ST_INTEGRATED", "ST_FRONTACCESSED", "ST_WDMSTANDARD", "ST_WDMEXTENDED", "ST_BOX", "ST_EXBOX", "ST_OSN1500A", "ST_OSN1500B", "ST_OSN3500T", "ST_INVALID"	
PreConfig	网元预配置信息	FIXED STRING， 取值范围： “0”, “1”	只有 WDM 子架支持该属性

7.7.6 SubnetworkConnection_T

字段名称	字段含义说明	取值说明	备注
LSPType	LSP 类型	FIXED STRING， 取值范围："LSP", "FA_LSP" 分别表示是普通 LSP 类型和服务 LSP 类型	仅对智能 SNC 有效
RelatedLsp	关联的 LSP	FIXED STRING	仅对智能 SNC 有效
RevertiveMode	恢复模式	FIXED STRING， 取值范围："Revertive", "Non-Revertive" 分别表示重路由可恢复和不可恢复	仅对智能 SNC 且该 SNC 不是无保护类型业务或额外业务时，该字段才有效。

7.7.7 SNCCreateData_T

字段名称	字段含义说明	取值说明	备注
BundledSNC	主用电路的标识	FREE STRING， 取值为创建预制电路时，使用该字段标识所属的主用电路	创建预制电路时需要填写该参数
Prefab	是否预制电路	FIXED STRING， 取值范围："true"， "false" 分别表示预制电路和非预制电路	
LSPType	LSP 类型	FIXED STRING， 取值范围："LSP"，"FA_LSP" 分别表示是普通 LSP 类型和服务 LSP 类型	
A1_Timeslot	指定源端时序	FREE STRING， 取值为创建电路时，指定的低阶时隙号[1-63]	该字段只有在非全路由方式创建汇聚型的 SNC 且输入的 CTP 信息中没有指定低阶时隙号的情况下才有效
Z1_Timeslot	指定宿端时序	FREE STRING， 取值为创建电路时，指定的低阶时隙号[1-63]	该字段只有在非全路由方式创建汇聚型的 SNC 且输入的 CTP 信息中没有指定低阶时隙号的情况下才有效

7.7.8 SNCModifyData_T

字段名称	字段含义说明	取值说明	备注
RevertiveMode	恢复模式	FIXED STRING， 取值范围："Revertive", "Non-Revertive" 分别表示重路由可恢复和不可恢复	仅对智能 SNC 且该 SNC 不是无保护类型业务或额外业务时，该字段才有效
ReroutePriority	重路由 LSP 的优先级	FIXED STRING， 取值范围："Low", "High" 分别表示重路由的优先级	仅对智能 SNC 有效

7.7.9 ProtectionSubnetwork_T

字段名称	字段含义说明	取值说明	备注
VC4Number	VC4 通道号	FREE STRING	该字段用于描述保护子网所占用的 VC4 通道

7.7.10 CrossConnect_T

字段名称	字段含义说明	取值说明	备注
Direction	表示交叉连接用作 SDH 的路径正向还是反向上	FIXED STRING 取值范围： "Reverse", "Obverse" 分别表示反向和正向	
ProtectionRole	指明该交叉用途	FIXED STRING 取值范围："Work", "Protection" 分别表示工作和保护	主要用于全路由创建电路时用来区分工作路由和保护路由。

7.7.11 HW_EthService_T

字段名称	字段含义说明	取值说明	备注
snkNodeNo	宿节点号	FREE STRING	仅当创建 RPR 相关的 EVPL 业务时需要填写该字段，否则业务无法创建成功。
svlan	QinQ 业务标签	FIXED STRING， 取值范围为整数	QinQ 业务需要使用该字段来标识多层标签

7.7.12 EthernetOAMOperation_T

字段名称	字段含义说明	取值说明	备注
srcCCActive	源端 CC 检测使能标识	FIXED STRING， 取值范围：" on", "off" 用于标识源端 CC 检测是否使能。	仅对 CC 检测操作才有意义
snkCCActive	宿端 CC 检测使能标识	FIXED STRING， 取值范围：" on", "off" 用于标识宿端 CC 检测是否使能。	仅对 CC 检测操作才有意义

7.7.13 ELLinkCreateData_T

字段名称	字段含义说明	取值说明	备注
ActivateStatus	ELL 的激活状态	FIXED STRING， 取值范围： "ACTIVE" , "DEACTIVE" 分别表示激活和未激活。	创建 ELL 时，用于指定 ELL 激活状态
Direction	ELL 的方向	FIXED STRING， 取值范围："CD_UNI", "CD_BI" 分别表示单向和双向	创建 ELL 时，用于指定 ELL 的方向

7.7.14 FDFrCreateData_T

字段名称	字段含义说明	取值说明	备注
ActivateStatus	FDFr 的激活状态	FIXED STRING， 取值范围： "ACTIVE" , "DEACTIVE" 分别表示激活和未激活。	创建 FDFr 时，用于指定 FDFr 激活状态
Direction	FDFr 的方向	FIXED STRING， 取值范围："CD_UNI", "CD_BI" 分别表示单向和双向	创建 FDFr 时，用于指定 FDFr 方向
EPLan	是否创建 EPLan 业务	FIXED STRING， 取值范围："1", " 0" 表示是否创建 EPLan 业务	只有在指定了 2 个 mac 口时，该字段才有效

7.7.15 NT_ALARM

字段名称	字段含义说明	取值说明	备注
AlarmSerialNo	告警产生的网管序列号	FREE STRING， 例如：" 10"	
AlarmReason	告警产生的原因	FREE STRING， 例如："接收线路侧信号丢失"	
ProductName	产生告警的设备类型	FREE STRING， 例如："OptiX 2500+"	
EquipmentName	产生告警的设备名称	FREE STRING， 例如：" SL4"	
AffirmState	告警确认状态	FREE STRING， 例如：”FALSE”	
DetailInfo	告警参数信息	FREE STRING， 例如：”光口号：1 通道号：1” 空值代表无额外告警参数信息	

7.7.16 NT_TCA

字段名称	字段含义说明	取值说明	备注
ProductName	产生 TCA 的设备类型	FREE STRING 例如："OptiX 2500+"	
EquipmentName	产生 TCA 的设备名称	FREE STRING 例如：" SL4"	

7.7.17 NT_PROTECTION_SWITCH

字段名称	字段含义说明	取值说明	备注
ProductName	发生倒换的设备类型	FREE STRING 例如：" OpitX Metro 1000V3"	
protectedEquipmentName	保护单板名称	FREE STRING 例如：" N1SL16"	
switchAwayFromEquipmentName	倒换源单板名称	FREE STRING 例如：" N1SL16"	
switchToEquipmentName	倒换宿单板名称	FREE STRING 例如：" N1SL16"	

7.7.18 NT_EPROTECTION_SWITCH

字段名称	字段含义说明	取值说明	备注
ProductName	发生倒换的设备类型	FREE STRING 例如：" OptiX OSN 3500	
protectedEquipmentName	保护单板名称	FREE STRING 例如：" LWF"	
switchAwayFromEquipmentName	倒换源单板名称	FREE STRING 例如：" LWF"	
switchToEquipmentName	倒换宿单板名称	FREE STRING 例如：" LWF"	

7.7.19 NT_WDMPROTECTION_SWITCH

字段名称	字段含义说明	取值说明	备注
ProductName	发生倒换的设备类型	FREE STRING 例如：" OptiX BWS1600G"	
protectedEquipmentName	保护单板名称	FREE STRING 例如：" LWF"	
switchAwayFromEquipmentName	倒换源单板名称	FREE STRING 例如：" LWF"	
switchToEquipmentName	倒换宿单板名称	FREE STRING 例如：" LWF"	

7.8 接口限制和约束说明

T2000 CORBA 接口存在以下限制和约束。

7.8.1 约束与限制 1

如果使用 CORBA 接口的 FlowDomain 模块，除了必须拥有 CORBA 接口相关 License 外，还必须拥有 SDH 和以太网端到端 License，即 License 包含 SupportSDHEndToEnd = 1 和 SupportIPEndToEnd = 1 。

7.8.2 约束与限制 2

对于 4.0 设备，CORBA 接口不能主动上报以下两类倒换事件。

- MSP 倒换事件
- SNCP 倒换事件

OSN 设备可以主动上报这两类倒换事件，但是 1:N 保护不支持自动上报。

7.8.3 约束与限制 3

CORBA 接口不能处理 ADL4 单板性能。

7.8.4 约束与限制 4

创建电路 createSNC 接口有以下限制：

SDH 领域

- createSNC 和 createAndActiveSNC 接口不支持创建跨智能域的电路。

- 目前静态保护级别的计算策略和努力程度（protectionEffort）参数只能为可接受任何级别（EFFORT_WHATEVER）。
- 如果要创建低阶 SNC，例如 E1 或 E3，必须首先创建 VC4 服务层路径。
- 如果指定的约束条件中包含了高阶时隙，这个约束条件只能用于创建服务层路径或高阶 SNC；如果约束条件中包含的是低阶时隙，这个约束条件只能用于创建低阶的 SNC。
- 支持全路由方式和非全路由方式，非全路由方式创建 SNC 只支持创建 ST_SIMPLE、ST_ADDROP_A、ST_ADDROP_Z 类型。
- 创建单点 ST_SIMPLE 型 SNC 时，只支持全路由方式（fullRoute 这个参数必须输入为 true），可以不用指定必经交叉（ccInclusions 这个参数，即可以输入指定的交叉连接，也可以什么也不输入）。如果创建的是非 ST_SIMPLE 型的单点 SNC 时，必须指定必经交叉（ccInclusions 参数必须输入指定的交叉连接）。
- 以全路由方式创建 SNC 时要把从源端到宿端经过的每个网元的交叉都在参数 ccInclusions 中描述出来。

WDM 领域

- 目前 WDM SNC 只支持创建 GE 速率的 Client 路径。
- 因为创建 SNC 时，需要指定的是可动态配置的交叉，所以波分 SNC 创建之前必须先连接光纤，进行波分路径搜索，得到 OCH 服务层路径。
- 不支持 userLabel 的唯一性（参数 forceUniqueness 数只能为 false）。
- 目前不支持重路由（参数 rerouteAllowed 参数只能为 RR_NA）。
- 目前只能支持 ST_SIMPLE 类型的 SNC 创建。
- 目前只支持单向类型 SNC 的创建。
- 目前不支持指定必经、必不经限制条件。

7.8.5 约束与限制 5

修改电路 modifySNC 接口有以下限制：

SDH 领域

- 目前只支持对 ST_SIMPLE 类型进行修改，源宿的端口、时隙不能更改。
- 只支持全路由方式修改（参数 fullRoute 必须填写 true）。
- 将 SNCP 业务转换成普通业务时，必须是全路由去保护的修改类型（参数 modifyType 必须输入 remove_protection，参数 fullRoute 必须输入 true）。
- 在 TMF 建议中，定义 modifyType 的取值为 add_protection/ remove_protection，分别对应将普通业务转换成 SNCP 业务和将 SNCP 业务转换成普通业务。必须在接口的 addedOrNewRoute/removedRoute 参数中输入要增加/删除的路由信息。

WDM 领域

- SNC 修改之前必须已经具有存在的 SNC；并且具有可动态配置的交叉。
- 不支持 userLabel 的唯一性（参数 forceUniqueness 该参数只能为 false）。
- 目前 SNC 只支持修改源、宿端（参数 modifyType 只能为 rerouting）。

- 目前只能支持 ST_SIMPLE 类型的 SNC 修改。
- 目前 SNC 修改只支持单向类型。

7.8.6 约束与限制 6

创建 ELL 接口 createELLink 接口（创建样例及具体应用说明可参考附录 C）有以下限制：

- 不支持 userLabel 的唯一性（参数 forceUniqueness 只能为 false）；
- 速率级别必须对应（参数 rate 只能为 LR_Encapsulation（98））；
- 目前不支持 RPR 环（参数 type 只能为 LT_POINT_TO_POINT）；
- 只能创建双向的 ELL；
- 未终结和单点 ELL 只支持手工创建服务层路径。

7.8.7 约束与限制 7

createFDFr 接口（创建样例及具体应用说明可参考附录 C）有以下限制：

- 端到端以太业务只支持使用已有的服务层路径创建，不支持自动创建服务层路径和手工创建服务层路径。
- 不支持 userLabel 的唯一性（参数 forceUniqueness 只能为 false）。
- 创建 FDFr 必须是未锁定的（参数 administrativeState 必须为 AS_Unlocked）。
- 连接需求目前只能为可忽略（参数 connectivityRequirement 只能为 CR_IGNORE）。
- 如果创建未终结和单点 FDFr，则不支持 QinQ 和 EPLan。
- 目前不能创建 EVPLan 业务。
- 目前只能修改 FDFr 的源宿内外部端口的端口属性（参数 tpsToModify 只能为源宿内外部端口的端口属性）。

7.8.8 约束与限制 8

波分 1:N 保护倒换上报有以下限制：

当 1：N 保护倒换事件是从保护恢复到工作时,上报的 FromTP 和 ToTP 数据不准确。

7.8.9 约束与限制 9

打开 BT 命名开关后，ELL 创建不支持自动创建服务层路径。

7.9 对象命名规则

本节依照 TMF 建议的命名规则举例介绍了 T2000 CORBA 接口中各种对象的命名规则。包括如下对象：

- EMS
- Subnetwork
- TopoSubnetwork
- ProtectionSubnetwork
- SubnetworkConnection
- ManagedElement
- TopologicalLink
- EPGP
- PTP
- CTP
- TrafficDescriptor
- EquipmentHolder
- Equipment
- ProtectionGroup
- WDM ProtectionGroup
- VirtualBridge
- VLAN
- Ethernet Service
- ATM Service
- ATM ProtectGroup
- QoS Rule
- Flow
- Flow Domain
- FlowDomainFragment
- EncapsulationLayerLink
- LinkAggregationGroup
- RPRNode
- Routing Area
- SNPPLink

7.9.1 EMS

对象名称	EMS
TMF 所建议的参数命名规则	name="EMS"; value="CompanyName/EMSname"
T2000 Corba 接口中的参数名称举例	name="EMS"; value="Huawei/T2000"
备注	如果需要同时管理多个 iManager T2000，则可以通过配置文件修改 T2000 的名称，以保证整个 NMS 管理域内每个 T2000 名称的唯一性。

7.9.2 Subnetwork

对象名称	Subnetwork
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="MultiLayerSubnetwork"; value="SubnetworkName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="MultiLayerSubnetwork"; value="1"
备注	

7.9.3 TopoSubnetwork

对象名称	TopoSubnetwork
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="TopoSubnetwork"; value=" TopoSubnetwork Name"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="TopoSubnetwork"; value="1"
备注	

7.9.4 ProtectionSubnetwork

对象名称	ProtectionSubnetwork
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ProtectionSubnetwork"; value="ProtectionSubnetworkName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ProtectionSubnetwork "; value="48"
备注	

7.9.5 SubnetworkConnection

对象名称	SubnetworkConnection
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="MultiLayerSubnetwork"; value="SubnetworkName" 3.name="SubnetworkConnection"; value="SubnetworkConnectionName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="MultiLayerSubnetwork"; value="25" 3.name="SubnetworkConnection"; value="123"
备注	

7.9.6 ManagedElement

对象名称	ManagedElement
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825"
备注	

7.9.7 TopologicalLink

对象名称	TopologicalLink
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="TopologicalLink"; value="TopologicalLinkName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="TopologicalLink"; value="345"
备注	

7.9.8 EPGP

对象名称	EPGP
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name=" EPGP"; value=" EPGPName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="EPGP"; value="589825/1/2/1"
备注	EPGP 值"589825/1/2/1"，第一位"589825"表示网元 ID，第二位"1"表示子架 ID，第三位"2"表示 EPGP 的 ID，第四位"1"表示扩展 ID。

7.9.9 PTP

对象名称	PTP	
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="PTP"; value="PTPName"	
T2000 Corba 接口中的参数名称举例	SDH	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="PTP"; value="/rack=1/shelf=1/slot=5/domain=sdh/port=2"
	WDM	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="33554433" 3.name="PTP"; value="/rack=1/shelf=590225/slot=8/domain=wdm/port=3"

对象名称	PTP	
	MSTP	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="PTP"; value="/rack=1/shelf=1/slot=5/domain=eth/type=mp/port=1"
备注	PTP 名字中 rack、shelf 的命名规则请参见 EquipmentHolder 部分。	
	MSTP	MSTP 领域 PTP，对于端口类型【type】的可能取值： 1.以太网域 mac：以太网外部端口 mp：以太网内部端口（VCTrunk 端口） rpr：以太网弹性分组环 RPR 端口 lp：VB 逻辑端口，例如 PTP 的 value="/rack=1/shelf=1/slot=5/domain=eth/type=lp/vb=1/port=1" 2.ATM 域（此时 domain=atm） atm：ATM 外部端口 atmtrunk：ATM 内部端口

7.9.10 CTP

对象名称	CTP	
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="PTP"; value="PTPName" 4.name="CTP"; value="CTPName"	
T2000 Corba 接口中的参数名称举例	SDH	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="PTP"; value="/rack=1/shelf=1/slot=5/domain=sdh/port=2" 4.name="CTP"; value="/sts3c_au4-j=1/vt2_tu12-k=1-l=3-m=2"
	WDM	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value= "12002" 3.name= "PTP"; value="/rack=1/shelf=12002SR1/slot=2/domain=wdm/port=1" 4.name="CTP"; value="/och=1/dsr=2"
	MSTP	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="590174" 3.name="PTP"; value="/rack=1/shelf=1/slot=4/domain=eth/type=mac/port=1" 4.name="CTP"; value="/tunnellabel=16/vclabel=0"

对象名称	CTP	
备注	SDH	1.对于 VC3 CTP，名字格式为 value="/sts3c_au4-j=1/tu3_vc3-k=2"； 2.对于 VC4 CTP，名字格式为 value="/sts3c_au4-j=3"，不支持 VC2 CTP； 3.对于 PDH 端口包含的 CTP，E4 级别的 CTP，名字格式为 value="/sts3c_au4=1"； 4.对于 PDH 端口包含的 CTP，E3 级别的 CTP，名字格式为 value="/tu3_vc3=1"； 5.对于 PDH 端口包含的 CTP，E1 级别的 CTP 名字格式为 value="/vt2_tu12=1"。
	WDM	根据应用场景，WDM CTP 可分为以下 6 种格式： 1.name="CTP"; value="/dsr=1"对应的应用场景如业务汇聚板、双向波长转换板的客户侧端口； 2. name="CTP"; value="/dsr=1/och=1"对应的应用场景只会出现在单向波长转换板上； 3. name="CTP"; value="/och=1/dsr=1"对应的应用场景如业务汇聚板的线路侧端口； 4. name="CTP"; value="/och=1"对应的应用场景如双向波长转换板、业务汇聚板的线路侧端口； 5. name="CTP"; value="/oms=1"对应的应用场景如合波分波板、分查复用板的线路侧端口。 6.name="CTP"; value="/os=1"对应的应用场景是光放板和光衰板，光线路保护板（OLP）等单板。
	MSTP	以太业务 CTP 划分方法是 PTP 加上划分流点的标签，各种标签输入可以为以下 3 种格式： 1.name="CTP"; value="/tunnellabel=16/vclabel=0"对应场景为 MPLS 端口，tunnellabel 不能为 0； 2.name="CTP"; value="/ethvid=0"对应场景为普通端口，ethvid＝0 独占，ethvid > 0 vlan 共享； 3.name="CTP"; value="/ethcvid=2/ethsvid=0"对应场景为 QinQ 端口； 4.name="CTP"; value="/eth=1"对应场景为基于端口透传的 QinQ 业务。

7.9.11 TrafficDescriptor

对象名称	TrafficDescriptor
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="TrafficDescriptor"; value="TrafficDescriptorName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei /T2000" 2.name="ManagedElement"; value="589825" 3.name="TrafficDescriptor"; value="101"
备注	根据 TraifficDescriptor 的名称可以唯一定位到网元下一个流量描述符。

7.9.12 EquipmentHolder

对象名称	EquipmentHolder	
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="EquipmentHolder"; value="EquipmentHolderName"	
T2000 Corba 接口中的参数名称举例	SDH 网元	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="EquipmentHolder"; value="/rack=1/shelf=1/slot=5"
	DWDM 网元	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="33554433" 3.name="EquipmentHolder"; value="/rack=1/shelf=590225/slot=1"
备注	对于华为公司设备，"rack"值始终为 1。 T2000 CORBA 接口目前支持 rack、shelf、slot 三种 EquipmentHolder 对象，不支持 subshelf 和 subslot 对象。	

7.9.13 Equipment

对象名称	Equipment
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="EquipmentHolder"; value="EquipmentHolderName" 4.name="Equipment"; value="EquipmentName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="EquipmentHolder"; value="/rack=1/shelf=1/slot=5" 4.name="Equipment"; value="1"
备注	对于华为公司设备而言，"Equipment"值始终为 1，即一个槽位只能安装一块单板。

7.9.14 ProtectionGroup

对象名称	ProtectionGroup
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="PGP"; value="ProtectionGroupName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="PGP"; value="1/5/2"
备注	

7.9.15 WDM ProtectionGroup

对象名称	WDM ProtectionGroup
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="WDMPG"; value="WDMPGName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="33554433" 3.name="WDMPG"; value="/pgtype=1/shelf=590004/pgID=1"
备注	

7.9.16 VirtualBridge

对象名称	VirtualBridge
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="vb"; value="VBName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="vb"; value="/rack=1/shelf=1/slot=2/vb=1"
备注	

7.9.17 VLAN

对象名称	VLAN
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="vlan"; value="VlanName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="vlan"; value="/rack=1/shelf=1/slot=2/vb=1/vlan=1"
备注	

7.9.18 Ethernet Service

对象名称	ETH Service
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="EthService"; value="EthServiceName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="EthService"; value="1/4/1/2"
备注	

7.9.19 ATM Service

对象名称	ATM Service
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="AtmService"; value="ATMServiceName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="AtmService"; value="1"
备注	

7.9.20 ATM ProtectGroup

对象名称	ATM ProtectGroup
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="AtmPG"; value="ATMPGName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="AtmPG"; value="1"
备注	

7.9.21 QoS Rule

对象名称	QoS
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="QosRule"; value="QosName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="QosRule"; value="/rack=1/shelf=1/slot=1/qostype=car/qos=1"
备注	

7.9.22 Flow

对象名称	Flow
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="Flow"; value="FlowName "
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="589825" 3.name="Flow"; value="/rack=1/shelf=1/slot=1/flow=1"
备注	

7.9.23 Flow Domain

对象名称	FlowDomain
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="Flowdomain"; value="FlowDomainName "
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="Flowdomain"; value="1"
备注	目前 T2000 中只有一个流域。

7.9.24 FlowDomainFragment

对象名称	FlowDomainFragment
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="Flowdomain"; value=" FlowdomainName" 3.name="FlowdomainFragment "; value="FlowdomainFragmentName "
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="Flowdomain"; value="1" 3.name="FlowdomainFragment"; value="2005-12-22 16:46:07 -54"
备注	

7.9.25 EncapsulationLayerLink

对象名称	EncapsulationLayerLink
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="EncapsulationLayerLink "; value="EncapsulationLayerLinkName "
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="EncapsulationLayerLink"; value="2005-12-22 16:59:35 -1770"
备注	

7.9.26 LinkAggregationGroup

对象名称	LinkAggregationGroup
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name="LAG"; value="LAGName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="590174" 3.name="LAG"; value="262145"
备注	

7.9.27 RPRNode

对象名称	RPRNode
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="ManagedElement"; value="ManagedElementName" 3.name=" RPRNode"; value=" RPRNodeName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="ManagedElement"; value="590174" 3.name="RPRNode"; value=" /rack=1/shelf=1/slot=5/node=1"
备注	

7.9.28 Routing Area

对象名称	RoutingArea
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="RoutingArea"; value="RoutingAreaName"
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="RoutingArea"; value="1"
备注	

7.9.29 SNPPLink

对象名称	SNPPLink
TMF 所建议的参数命名规则	1.name="EMS"; value="CompanyName/EMSname" 2.name="RoutingArea"; value="RoutingAreaName" 3.name="SnppLink"; value="SnppLinkName "
T2000 Corba 接口中的参数名称举例	1.name="EMS"; value="Huawei/T2000" 2.name="RoutingArea"; value="1" 3.name="SnppLink"; value="1"
备注	

7.10 非功能互通性声明

7.10.1 EMS 子网配置

在 TMF MTNM 建议中，子网根据网络拓扑结构或网络逻辑分层结构对 EMS 所管理的网元进行的逻辑上的划分。原则上可以将整个 EMS 管理域看作一个子网，也可以细化到将一个网元、一个环、一条链等拓扑划分看作一个子网。至于如何划分，建议只是给出了参考的原则。由于 T2000 同时具有网元级和网络级的业务管理能力，在进行子网划分时如果划分粒度较小，会导致 OSS 对网络的管理复杂化。因此在 T2000V200R005C01B01X 版本，将整个 EMS 上的网元全部划分到一个子网中，子网类型为 MESH 型，子网 ID 固定为 1，以简化网络结构。

7.10.2 SNC 模式

在 TMF MTNM 建议中定义了 A、B、C、D 四种 SNC 状态模式，T2000 CORBA 接口对于这四种状态模式的支持情况如表 7-98 所示。

表7-98 SNC 状态模式

模式	支持未激活状态的 SNC，并允许创建共享交叉连接	支持激活状态的共享交叉连接
A		√
B		
C		
D	√	√

上表中 SNC 状态的四种模式在 TMF MTNM 建议有详细的描述，具体内容如下：

Mode A

In Mode A, the goal is for the EMS to represent only the current network configuration, to limit as much as possible sharing of resources (i.e., cross connections) among SNCs, and to attempt to have a one-to-one correspondence between the network configurations and the SNC configurations. This mode does not support the pending state. An SNC that does not have any non-shared, active CCs in the network is considered non-existent. Consequently, when the last non-shared cross-connect of an SNC is deactivated in the network, the SNC is deleted by the EMS.

In Mode A, a CC, active or not, can only belong to at most one SNC. In the case of non-singleton sub-networks, an exception to the non-sharing rule is made for SNCs of a broadcast system, i.e., a multipoint connection. The legs of a multipoint connection are represented as individual SNCs.

The legs of a broadcast system may share CCs and always share the same source CTP.

Mode A is best for EMSs that only support singleton sub-networks (i.e., sub-network consisting of a single managed element) and that do not keep a database of pending SNCs. Mode A might be popular with vendors having lightweight switch-bound EMSs.

Mode B

In Mode B, the goal is for the EMS to represent the current network configuration as well as potential "future" SNCs that have been prepared by the NMS, but not yet activated. The Pending state can also be used in situations where SNC share CCs at different times. For example, SNC1 is used by customer A from 8am to 8pm every day. SNC2 shares many CCs with SNC1 but is only used by Customer B from 9pm to 7am every day. When SNC1 is in the Active state, SNC2 is put in the Pending state, and vice versa.

An SNC’s entry and exit to and from the Pending state is controlled solely by the NMS. Neither the EMS or craft intervention can put an SNC into (or remove an SNC from) the Pending state.

Mode C

In Mode C, the goal is for the EMS to represent only the current network configuration (as was the case with Mode A). Contrary to Mode A, it does not limit sharing of CCs among SNCs, and does not attempt to have a one-to-one correspondence between the network configurations and the SNC configurations.

Mode C is best for EMSs that support non-singleton sub-networks and that do not keep a database of pending SNCs. This mode has an advantage over Mode A, because it allows SNC

reorganizations without traffic interruption (only useful in non-singleton sub-networks). For example, if the EMS currently has two "consecutive" SNCs that the NMS wants to merge into one "larger" SNC, this can be done without interrupting traffic by creating and activating the larger SNC, then deactivating and deleting the two consecutive SNCs.

Mode D

Mode D is basically a combination of Modes B and C. This Mode is favored by vendors (or service providers) that have a feature rich NMS that can take advantage of the Pending state (related to scheduling features) and the sharing of Active CCs.

The T2000 CORBA interface supports two types of SNCs that correspond to the circuits in the T2000. One is normal SNC and the other is preconfigured SNC. The supported SNC states are different for the two types of SNCs.

T2000 CORBA 接口支持两种类型的 SNC（和 T2000 网管中的电路相对应），一种为普通的 SNC；另一种为预制 SNC。对于这两种不同的 SNC，支持的 SNC 模式也是不同的：

- 对于普通的 SNC，支持未激活、部分激活、激活三种状态，但无论在何种状态下，都不支持资源共享，即一条交叉连接同一时刻只能属于一条 SNC。
- 对于预制 SNC，支持未激活、部分激活、激活三种状态。同时支持资源共享，即一条交叉连接可以同时为多条 SNC 共享，但在同一时刻只能属于一条激活的 SNC。

7.10.3 迭代器使用

T2000 CORBA 接口完全支持 TMF814 中定义的各种迭代器（Iterator），提供了一整套迭代器的管理方法，能保证迭代器资源在任何情况下（正常和异常）都能得到正确的释放，防止内存泄漏。

T2000 CORBA 接口支持迭代器数量限制功能，接口缺省最多同时能支持迭代器个数为 20（该数值可以根据需要配置其大小，用法参考 5.4.2　T2000 配置工具的安装和配置）。当迭代器个数超过最大限制时，T2000 CORBA 接口会自动销毁前面最先生成的迭代器。借此 T2000 可以有效控制系统负荷，防止系统崩溃。

在一定情况下会生成迭代器对象（例如接口查询结果个数大于输入的 how_many 参数时，会产生迭代器对象），T2000 CORBA 接口提供的迭代器功能支持 TMF814 中定义的三种标准操作：

- destroy：销毁该迭代器（Iterator）对象。
- getLength：获取迭代器中记录条数。迭代器中的记录条数 = 总的记录数 – how_many 参数值，而且该记录数在迭代器生命周期内是不会发生变化的。
- next_n：获取下一批数据，最多返回 n 条记录。当迭代器中还有数据未取完时，next_n 方法会返回 true；当迭代器中的所有数据都已经被取完时，next_n 方法会返回 false，同时 T2000 CORBA 接口也会自动销毁该迭代器对象，这种情况下，上层网管不需要再显式的调用 destroy 方法来销毁该迭代器。

7.10.4 心跳检测机制

心跳检测机制用于检测 T2000 CORBA 接口和高层网管的连通性。目前 T2000 CORBA 接口支持两种心跳检测机制：

- ping 操作
- 心跳通知

ping 操作

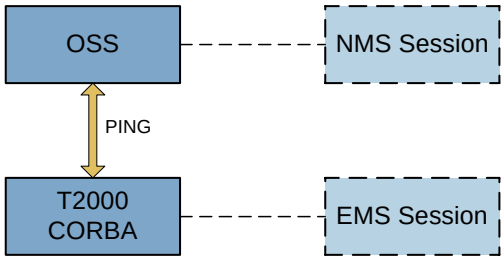
当高层网管系统在向 T2000 CORBA 接口发起连接请求时，除了输入登录帐号信息外，上层网管还需要输入一个 NMS Session_I 对象的对象引用。请求成功后，上层网管和 T2000 CORBA 接口之间会建立起一条会话连接，具体而言，就是 T2000 CORBA 接口会返回一个 EMSSession_I 对象的对象引用给上层网管，可参考接口原型定义：

```
void getEmsSession (in string user, in string password, in
nmsSession::NmsSession_I client, out emsSession::EmsSession_I
```

emsSessionInterface) raises (globaldefs::ProcessingFailureException)

两者通过互相调用 ping 来测试连通性，如图 7-3 所示。

图7-3 ping 操作

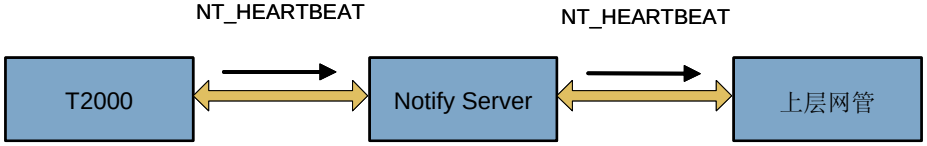


会话连接建立后，T2000 CORBA 接口会定时回调上层网管注册对象 NMSSession_I 的 ping 方法，以检测上层网管是否运行正常以及网络通信连接是否正常。如果连续 4 次 ping 操作都返回失败，T2000 CORBA 接口会自动中断和上层网管的会话连接并释放相应的会话资源。

心跳通知

ping 操作主要用于检测 T2000 CORBA 接口和上层网管的通信连接状态以及运行状态，对于通知服务的服务状态检测则主要依靠心跳通知来完成。因此，采用心跳通知上报来检测通知服务。这是一个单项的检测机制，T2000 的 CORBA 接口定时（默认为 30 秒）向通知服务器发送一条 NT_HEARTBEAT 通知，Notify Server 再发送给上层网管，这样上层网管可以周期性的收到心跳通知，保证了通知服务和上层网管之间的连通性。

图7-4 心跳通知



7.10.5 BT 命名格式使用说明

BT 命名规则使用场合与原则

BT 命名规则是 T2000 CORBA 接口为适应上层网管特殊应用场景的要求而实现的直接采用对象的名字作为接口中对象唯一标识的一种方法。它主要针对 ManagedElement 等常用的实体对象，不再采用网管内部 ID 作为对象的唯一标识，而是改为直接采用对象的名字（即 T2000 网管界面显示的实体对象名称）来作为对象的唯一标识（支持的具体对象及示例，请参考 7.10.5 BT 命名格式使用说明）。

由于 T2000 网管界面中显示的实体对象名称并不一定保证其在网管内的唯一性，因此如果上层网管系统希望 T2000 CORBA 接口使能 BT 命名规则，则上层网管系统必须通过网络资源的合理规划等手段来确保相关的实体对象的名字在 T2000 网管中唯一。这种唯一性上层网管必须保证，否则会导致 T2000 CORBA 接口上报的对象数据出现冲突或混乱。

BT 命名规则的使用方法

步骤 1 在网管启动之前将 %IMAP%目录下配置文件 corba.cfg 中 bUseBTName = 0 修改为 bUseBTName = 1，并保存该配置文件。

步骤 2 启动网管及 CORBA 接口（注：启动方法参考"第 5 章系统安装和运行"）。

步骤 3 调用 T2000 CORBA 接口提供的方法时，如果方法输入参数中需要输入请求操作的实体对象的标识，且被请求的实体对象是 BT 命名规则支持的对象，则需要输入 BT 命名规则格式的对象标识，而不是未使能 BT 命名规则时的对象标识。

----结束

BT 命名格式支持的对象及示例

表7-99 ManagedElement 对象

开/关	说明	示例	备注
关	接口上报或查询使用 ME 唯一编号形式	**光网元：** {name EMS value Huawei/T2000} {name ManagedElement value 33554433}	• 根据对象标识的层次模型关系，ManagedElement 对象的命名规则改变会影响的子对象包括：EquipmentHolder(rack/shelf/slot)、Equipment、PTP、CTP、VB、VLAN、CAR、COS、Flow、TrafficDescriptor 等。即所有对象标识被 ManagedElement 对象包含的实体对象的命名规则会相应的受到影响。 • 对象命名规则请参考《对象命
		非光网元： {name EMS value Huawei/T2000} {name ManagedElement value 590174}	
开	接口上报或查询使用 BT 命名方式，该方	**光网元：** {name EMS value Huawei/T2000} {name ManagedElement value otm1}	

开/关	说明	示例	备注
	式与 T2000 前台显示的 ME 名称一致	**非光网元：** {name EMS value Huawei/T2000} {name ManagedElement value NE350}	名》章节。

说明

开/关 指当前是否使用 BT 命名规则，方法详见 7.10.5 BT 命名格式使用说明，下同。

表7-100 SubnetworkConnection 对象

开/关	说明	示例	备注
关	接口上报或查询使用 SNC 唯一编号形式	{name EMS value Huawei/T2000} {name MultiLayerSubnetwork value 1} {name SubnetworkConnection value {2005-12-20 14:26:22 - 50}}	
开	接口上报或查询使用 BT 命名方式，该方式与 T2000 前台显示的 SNC 名称一致	{name EMS value Huawei/T2000} {name MultiLayerSubnetwork value 1} {name SubnetworkConnection value NE350-NE351-VC12-0001}	

表7-101 EncapsulationLayerLink 对象

开/关	说明	示例	备注
关	接口上报或查询使用 ELL 唯一编号形式	{name EMS value Huawei/T2000} {name EncapsulationLayerLink value {2006-03-21 16:01:22 - 1134}}	
开	接口上报或查询使用 BT 命名方式，该方式与 T2000 前台显示的 ELL 名称一致	{name EMS value Huawei/T2000} {name EncapsulationLayerLink value Trunk_Link-0001}	

表7-102 FlowDomainFragment 对象

开/关	说明	示例	备注
关	接口上报或查询使用FDFr 唯一编号形式	{name EMS value Huawei/T2000} {name Flowdomain value 1} {name FlowdomainFragment value {2005-12-22 16:46:07 - 54}}	
开	接口上报或查询使用BT 命名方式，该方式与 T2000 前台显示的 FDFr 名称一致	{name EMS value Huawei/T2000} {name Flowdomain value 1} {name FlowdomainFragment value EVPL-0001}	

表7-103 TopologicalLink 对象

开/关	说明	示例	备注
关	接口上报或查询使用TopoLogicalLink 唯一编号形式	{name EMS value Huawei/T2000} {name TopologicalLink value {2006-03-21 15:37:56 - 723}}	
开	接口上报或查询使用BT 命名方式，该方式与 T2000 前台显示的TopoLogicalLink 名称一致	{name EMS value Huawei/T2000} {name TopologicalLink value NE350-NE351-MS-0001}	

表7-104 EquipmentHolder 对象

开/关	说明	示例	备注
关	接口上报或查询使用EquipmentHolder 唯一编号形式	**光网元：** {name EMS value Huawei/T2000} {name ManagedElement value 33554433} {name EquipmentHolder value /rack=1/shelf=590004/slot=1}	根据对象唯一标识的层次模型关系，Equipment Holder 对象命名规则会影响挂接在该对象下的 Equipment 对象的命名规则； 对象命名规则请参考 7.9 对象命名规则。
		非光网元： {name EMS value Huawei/T2000} {name ManagedElement value 590174} {name EquipmentHolder value /rack=1/shelf=1/slot=1}	

开/关	说明	示例	备注
开	接口上报或查询使用 BT 命名方式，该方式与 T2000 前台显示的 EquipmentHolder 名称一致	**光网元：** {name EMS value Huawei/T2000} {name ManagedElement value otm1} {name EquipmentHolder value /rack=1/shelf=NE180/slot=1}	
		非光网元： {name EMS value Huawei/T2000} {name ManagedElement value NE350} {name EquipmentHolder value /rack=1/shelf=1/slot=1}	

7.10.6 SSL

SSL 原理

SSL（Security Socket Layer）的中文意思是"安全套接层协议"，它是由 Netscape 公司推出的一种安全通信，通过使用数字证书 SSL 可以让客户机和服务器之间建立一条加密的安全通道，能保证所传输的信息不被他人非法窃取。

当采用了 SSL 加密机制后，客户机首先要与服务器建立通信连接，服务器会把数字证书和公开密钥发送给客户机，与客户机交换密码，一般选用的是 RSA 密码算法（也有选用 Diffie-Hellman 和 Fortezza-KEA 密码算法）。当身份验证确认后，这个公开密钥对用户端的会话密钥进行加密并将其传送到服务器，服务器接到这个会话密钥后会对会话密钥进行解密，这时用户就和服务器建立了一条加密通信通道。

SSL 的三要素包括：证书、数字签名、加密。关于 SSL 的概念和详细原理请参考相关的书籍和资料，这里不做详细描述。

CORBA 接口对 SSL 的支持

SSL 由 T2000 的 CORBA 接口和上层网管通讯时，支持两种协议模式：IIOP 、SSLIOP。CORBA 接口和上层网管的加密通讯只支持 SSLIOP 模式

SSLIOP 模式：采用 SSLIOP 需要根据证书处理进行加密，要求双方采用的证书要相同。这种模式下，要求上层网管必须使用 SSL 方式和 T2000 CORBA 接口进行通信，即必须使用证书、数字签名、加密算法等手段来保证系统之间通信的安全性。根据实际需要, 可以通过修改 T2000 CORBA 接口配置参数来选择是否需要对交互的数据进行加密。同时，用户也可以根据需要自己管理证书。

SSL 的配置

T2000 CORBA 接口对 SSL 协议的支持，可以通过修改启动配置实现。修改方法可参考"第 5 章　系统安装和运行"。