This appendix lists interceptors and use cases to resynchronize the NMS from the EMS. This appendix includes the following sections:

- C.1 Use Cases to Resynchronize the NMS from the EMS, page C-1
- C.2 Developing a GateWay/CORBA Client, page C-4

### C.1 Use Cases to Resynchronize the NMS from the EMS

Table C-1, Table C-2, and Table C-3 list use cases to resynchronize the NMS from the EMS.

<table>
<thead>
<tr>
<th>Information</th>
<th>Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The NMS retrieves Cisco Prime Optical information.</td>
</tr>
<tr>
<td>Summary</td>
<td>The NMS retrieves all EMS and ME alarms and information for existing multilayer subnetworks, MEs, PTPs, CTPs, SNCs, and protection groups from Cisco Prime Optical GateWay/CORBA interfaces.</td>
</tr>
<tr>
<td>Actor(s)</td>
<td>NMS.</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The NMS locates the EmsSessionFactory object and obtains references to EMSMgr_I, MultiLayerSubnetworkMgr_I, ManagedElementMgr_I, EquipmentInventoryMgr_I, MaintenanceMgr_I, ProtectionMgr_I, and PerformanceManagementMgr_I interfaces.</td>
</tr>
<tr>
<td>Begins When</td>
<td>The NMS sends a request to retrieve all of the Prime Optical information.</td>
</tr>
</tbody>
</table>
C.1 Use Cases to Resynchronize the NMS from the EMS

Description

1. Uses the emsMgr::EMSMgr_I::getAllEMSSystemActiveAlarms interface to request all EMS alarms.
2. Uses the emsMgr::EMSMgr_I::getAllTopLevelSubnetworks interface to request all multilayer subnetworks in Prime Optical.
   Prime Optical returns a list of MultiLayerSubnetwork_T objects with detailed information about a multilayer subnetwork.
3. Uses the emsMgr::EMSMgr_I::getAllTopLevelTopologicalLinks interface to request all topological links across multiple multilayer subnetworks.
   Prime Optical returns a list of TopologicalLink_T objects with detailed information about a topological link.
4. For each MultiLayerSubnetwork_T object, Prime Optical uses the multiLayerSubnetwork::MultiLayerSubnetworkMgr_I::getAllManagedElements interface to request all MEs that belong to that multilayer subnetwork.
   Prime Optical returns a list of ManagedElement_T objects with detailed information about an ME.
   For each ManagedElement object, executes the use case “NMS Resynchronizes Information Specific to a Managed Element.”
5. Uses the multiLayerSubnetwork::MultiLayerSubnetworkMgr_I::getAllVLANs interface to request all VLANs associated with the multilayer subnetwork.

Ends When
The NMS retrieves all Prime Optical information.

Exceptions
Refer to the exceptions thrown by the individual interface method.

Post Conditions
- The NMS synchronizes with Prime Optical.
- The NMS registers with Prime Optical to retrieve notifications related to changes in the managed object on Prime Optical, new alarms, and TCAs.

Traceability
—

Table C-2 NMS Resynchronizes Information Specific to a Managed Element

<table>
<thead>
<tr>
<th>Information</th>
<th>Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The NMS retrieves Prime Optical information specific to the ME.</td>
</tr>
<tr>
<td>Summary</td>
<td>The NMS retrieves all PTP, CTP, SNC, and protection group information for the specific ME.</td>
</tr>
<tr>
<td>Actor(s)</td>
<td>NMS.</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The NMS obtains references to ManagedElementMgr_I, EquipmentInventoryMgr_I, MaintenanceMgr_I, ProtectionMgr_I, and PerformanceManagementMgr_I interfaces. The NMS also obtains the ME name.</td>
</tr>
<tr>
<td>Begins When</td>
<td>The NMS sends a request to retrieve all information about an ME. The NMS identifies the name of the ME for which to retrieve information.</td>
</tr>
</tbody>
</table>
### Use Cases to Resynchronize the NMS from the EMS

**Description**
The NMS does the following to request information from the GateWay/CORBA interface:

1. Uses the `managedElementManager::ManagedElementManager_I::getAllActiveAlarms` interface to request all current alarms on the ME.
2. Uses the `equipment::EquipmentInventoryMgr_I::getAllEquipment` interface to request all existing equipment on the ME.
   
   Prime Optical returns a list of `EquipmentHolder_T` objects with detailed information about equipment holders and equipment on the ME.
3. For the equipment information retrieved in `EquipmentHolder_T` object, uses the `equipment::EquipmentInventoryMgr_I::getAllSupportedPTPs` interface to request all supported PTPs on the equipment.
   
   Prime Optical returns a list of `TerminationPoint_T` objects with detailed PTP information, such as SDH or SONET port, admin state, service state, line code, frame format, and so on.
4. For each `TerminationPoint_T` object, executes the “NMS Resynchronizes Information Specific to TerminationPoint” use case.
5. Uses the `managedElementManager::ManagedElementManager_I::getAllSNCs` interface to request all SNCs originating, terminating, or passing through the ME.
   
   Prime Optical returns a list of `SubnetworkConnection_T` objects with detailed SNC information, such as layer rate, SNC state, SNC protection state, a-end CTP, z-end CTP, and so on.
6. Uses the `protection::ProtectionMgr_I::getAllProtectionGroups` interface to request all protection groups created on the ME.
   
   Prime Optical returns a list of `ProtectionGroup_T` objects with detailed protection group information, such as 1:1, 1+1, or 2F-BLSR; revertive or nonrevertive; and so on.

**Ends When**
The NMS retrieves all ME-specific information.

**Exceptions**
Refer to the exceptions thrown by the individual interface method.

**Post Conditions**
The NMS synchronizes information about an ME with Prime Optical.

**Traceability**
---

### Table C-2 NMS Resynchronizes Information Specific to a Managed Element (continued)

<table>
<thead>
<tr>
<th>Information</th>
<th>Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The NMS does the following to request information from the GateWay/CORBA interface:</td>
</tr>
<tr>
<td></td>
<td>1. Uses the <code>managedElementManager::ManagedElementManager_I::getAllActiveAlarms</code> interface to request all current alarms on the ME.</td>
</tr>
<tr>
<td></td>
<td>2. Uses the <code>equipment::EquipmentInventoryMgr_I::getAllEquipment</code> interface to request all existing equipment on the ME.</td>
</tr>
<tr>
<td></td>
<td>Prime Optical returns a list of <code>EquipmentHolder_T</code> objects with detailed information about equipment holders and equipment on the ME.</td>
</tr>
<tr>
<td></td>
<td>3. For the equipment information retrieved in <code>EquipmentHolder_T</code> object, uses the <code>equipment::EquipmentInventoryMgr_I::getAllSupportedPTPs</code> interface to request all supported PTPs on the equipment.</td>
</tr>
<tr>
<td></td>
<td>Prime Optical returns a list of <code>TerminationPoint_T</code> objects with detailed PTP information, such as SDH or SONET port, admin state, service state, line code, frame format, and so on.</td>
</tr>
<tr>
<td></td>
<td>4. For each <code>TerminationPoint_T</code> object, executes the “NMS Resynchronizes Information Specific to TerminationPoint” use case.</td>
</tr>
<tr>
<td></td>
<td>5. Uses the <code>managedElementManager::ManagedElementManager_I::getAllSNCs</code> interface to request all SNCs originating, terminating, or passing through the ME.</td>
</tr>
<tr>
<td></td>
<td>Prime Optical returns a list of <code>SubnetworkConnection_T</code> objects with detailed SNC information, such as layer rate, SNC state, SNC protection state, a-end CTP, z-end CTP, and so on.</td>
</tr>
<tr>
<td></td>
<td>6. Uses the <code>protection::ProtectionMgr_I::getAllProtectionGroups</code> interface to request all protection groups created on the ME.</td>
</tr>
<tr>
<td></td>
<td>Prime Optical returns a list of <code>ProtectionGroup_T</code> objects with detailed protection group information, such as 1:1, 1+1, or 2F-BLSR; revertive or nonrevertive; and so on.</td>
</tr>
<tr>
<td>Ends When</td>
<td>The NMS retrieves all ME-specific information.</td>
</tr>
<tr>
<td>Exceptions</td>
<td>Refer to the exceptions thrown by the individual interface method.</td>
</tr>
<tr>
<td>Post Conditions</td>
<td>The NMS synchronizes information about an ME with Prime Optical.</td>
</tr>
<tr>
<td>Traceability</td>
<td>---</td>
</tr>
</tbody>
</table>
GateWay/CORBA implements APIs defined by the TeleManagement Forum’s Multi Technology Network Management (MTNM) group. These APIs are defined for communication between an NMS and the EMS. The NMS must develop a client application that uses these APIs. The following sections describe the tools required for developing a client application. Sample code is provided.

C.2.1 Recommended Tools

You can develop the CORBA client on UNIX or PC platforms. Cisco recommends the following development tools:

- Sun Microsystems Java Development Kit (JDK) 1.6 (available at http://java.sun.com/)
- jacORB 2.x
C.2.2 Sample Code in Java

A typical CORBA client application involves the following steps:

---

**Step 1** Initialize the client connection to the object request broker (ORB).

**Step 2** Obtain a reference to the naming service.

**Step 3** Resolve the reference to EMSSessionFactory. See C.2.2.3 Get Reference to EMSSessionFactory, page C-7.

**Step 4** Implement NmsSession_IOperations. See C.2.2.4 Implement NmsSession_IOperations, page C-7.

**Step 5** Retrieve EmsSession by supplying the username and password. See C.2.2.5 Log In and Retrieve EmsSession, page C-8.

**Step 6** Query EmsSession and obtain a list of managers available for operations. See C.2.2.6 Retrieve List of Managers, page C-9.

**Step 7** Invoke the desired method on that manager. See C.2.2.7 getEMS Operation on EMS Manager, page C-9.

If you want your client to receive notifications from GateWay/CORBA, the following additional steps are required:

**Step 8** Obtain a reference to an EventChannel object in the notification server in Step 2.

**Step 9** Obtain a ConsumerAdmin object reference.

**Step 10** Invoke obtain_notification_push_consumer() on the SupplierAdmin object, specifying CosNotifyChannelAdmin::STRUCTURED_EVENT as a parameter. This operation returns a reference to the StructuredProxyPushSupplier.

**Step 11** Implement an instance of StructuredPushConsumer (defined by the OMG).

**Step 12** Invoke the connect_structured_push_consumer() operation on the StructuredProxyPushSupplier object to connect the StructuredPushConsumer implementation object to the notification channel.

**Step 13** Monitor incoming notifications.

---

The following sections show sample Java code for the steps required to develop a client application.

**Note** The sample code shown is for illustrative purposes only; the code might not compile as shown. The sample code does not handle all exceptions.

C.2.2.1 Initialize the Client Connection

```java
// Import OMG packages
import org.omg.CORBA.IntHolder;
import org.omg.CORBA.ORB;
import org.omg.CORBA.Policy;

// Import naming context packages.
import org.omg.CosNaming.NameComponent;
import org.omg.CosNaming.NamingContextExt;

// Import notify channel and event service packages.
import org.omg.CosEventChannelAdmin.AlreadyConnected;
```
import org.omg.CosEventChannelAdmin.TypeError;
import org.omg.CosNotifyChannelAdmin.AdminLimitExceeded;
import org.omg.CosNotifyChannelAdmin.AdminNotFound;
import org.omg.CosNotifyChannelAdmin.ClientType;
import org.omg.CosNotifyChannelAdmin.ConsumerAdmin;
import org.omg.CosNotifyChannelAdmin.EventChannel;
import org.omg.CosNotifyChannelAdmin.EventChannelHolder;
import org.omg.CosNotifyChannelAdmin.ProxySupplier;
import org.omg.CosNotifyChannelAdmin.StructuredProxyPushSupplier;
import org.omg.CosNotifyChannelAdmin.StructuredProxyPushSupplierHelper;

// Import POA packages
import org.omg.PortableServer.POA;
import org.omg.PortableServer.POAHelper;
import org.omg.PortableServer.POAPackage.ServantAlreadyActive;
import org.omg.PortableServer.POAPackage.WrongPolicy;

// Import TMF packages
import org.tmforum.mtnm.emsSession.EmsSession_I;
import org.tmforum.mtnm.emsSession.EmsSession_IHolder;
import org.tmforum.mtnm.emsSession.EmsSession_IPackage.managerNames_THolder;
import org.tmforum.mtnm.emsSessionFactory.EmsSessionFactory_I;
import org.tmforum.mtnm.emsSessionFactory.EmsSessionFactory_IHelper;
import org.tmforum.mtnm.nmsSession.NmsSession_I;
import org.tmforum.mtnm.nmsSession.NmsSession_IPOATie;

public static void main(String[] args)
{
    try {
        // Optional: set up ORB properties: you can set them here or in the launch file
        // for the client
        // Properties sys_properties = System.getProperties();
        // For JacORB
        // sys_properties.put("org.omg.CORBA.ORBClass", "org.jacorb.orb.ORB"
        // sys_properties.put("org.omg.CORBA.ORBSingletonClass", "org.jacorb.orb.ORBSingleton"
        // For Orbix
        // sys_properties.put("org.omg.CORBA.ORBClass", "com.iona.corba.art.artimpl.ORBImpl");
        // sys_properties.put("org.omg.CORBA.ORBSingletonClass", "com.iona.corba.art.artimpl.ORBSingleton");
        // For Visibroker
        // sys_properties.put("org.omg.CORBA.ORBClass", "com.inprise.vbrokerORB.orb.ORB"
        // sys_properties.put("org.omg.CORBA.ORBSingletonClass", "com.inprise.vbrokerORB.orb.ORBSingleton"
        // For other ORBs check the related ORB programming guide

        // Step 1: Initialize the ORB and obtain ROOT POA reference
        // Note: ORB_init will process any -ORB arguments
        // be called before any other argument processing.
        //
        org.omg.CORBA/Object root_poa = global_orb.resolve_initial_references("RootPOA");
        POA rpoa = POAHelper.narrow(root_poa);
        POA poa = rpoa.create_POA("myPolicy",null, new Policy[0]);
poa.the_POAManager().activate();
}
catch (SystemException ex) {
    // Exception handling
}
}

C.2.2.2 Get Reference to the Naming Service

// Get Nameservice reference
NamingContext nsRootContext = null;
try {
    // Step 2: Get reference to the name service
    // Option 1: Resolve initial reference (RIR)
    // org.omg.CORBA.Object obj = global_orb.resolve_initial_references("NameService");
    // or
    // Option 2: corbaloc URL
    String objRef = "corbaloc:iiop:gatewayserver.cisco.com:14005/NameServiceGWC";
    org.omg.CORBA.Object obj = global_orb.string_to_object(objRef);
    /* NOTE: Please replace "gatewayserver.cisco.com" with the name of the server on
     * which CORBA Naming service is running. */
    // Narrow to root naming context
    NamingContextExt root_context = NamingContextExtHelper.narrow(obj);
}
catch (org.omg.CORBA.ORBPackage.InvalidName inEx) {
    // Exception handling
}
}

C.2.2.3 Get Reference to EMSSessionFactory

Follow the example in C.2.2.2 Get Reference to the Naming Service, page C-7 to obtain a reference to
the naming service.

NameComponent name = new NameComponent[6];
name[0] = new NameComponent("TMF_MTNM", "Class");
name[1] = new NameComponent("Cisco Systems", "Vendor");
name[2] = new NameComponent("Cisco Prime Optical", "EMSInstance");
name[3] = new NameComponent(version, "Version"); //where version = "9_5" for Prime Optical 9.5
name[4] = new NameComponent(ctm_sys_id, "EMS"); // ctm_sys_id = "PrimeOptical"
name[5] = new NameComponent("SessionFactory", "EmsSessionFactory");
try {
    org.omg.CORBA.Object emsSessionI = root_context.resolve(name);
}
catch (InvalidName inEx) {
    // Exception handling
}
catch (NotFound nfEx) {
    // Exception handling
}

C.2.2.4 Implement NmsSession_IOperations

import org.tmforum.mtnm.session.*;
import org.tmforum.mtnm.nmsSession.*;

public class SessionImpl implements NmsSession_IOperations {

C.2.2 Sample Code in Java

Session_I myAssociatedSession = null;

public SessionImpl() {
    super();
    // TODO Auto-generated constructor stub
}

public void setAssociatedSession(Session_I emsSession) {
    myAssociatedSession = emsSession;
}

public Session_I getAssociatedSession() {
    return myAssociatedSession;
}

public void eventLossOccurred(String startTime, String notificationId) {
    // TODO Auto-generated method stub
}

public void eventLossCleared(String endTime) {
    // TODO Auto-generated method stub
}

public void historyPMDataCompleted(String fileName) {
    // TODO Auto-generated method stub
}

public void historyPMDataFailed(String errorReason) {
    // TODO Auto-generated method stub
}

public Session_I associatedSession() {
    // TODO Auto-generated method stub
    return null;
}

public void ping() {
    // TODO Auto-generated method stub
}

public void endSession() {
    // TODO Auto-generated method stub
}

C.2.2.5 Log In and Retrieve EmsSession

To perform operations from GateWay/CORBA, your client must log in using a username and password created on the Prime Optical client. See B.1 Creating an OSS Client Profile for GateWay/CORBA, page B-1.

EmsSession_I m_emsSession = null;
SessionImpl mySessionImpl = new SessionImpl();
try {

}
EmsSessionFactory_I ems_ref = EmsSessionFactory_IHelper.narrow(emsSessionI);
EmsSession_IHolder emsSessionHldr = new EmsSession_IHolder();
NmsSession_IPOATie tieobj = new NmsSession_IPOATie(mySessionImpl, poa);
poa.activate_object(tieobj);
NmsSession_I nmsSession_ref = tieobj._this();
if (ems_ref != null) {
    ems_ref.getEmsSession(user, password, nmsSession_ref, emsSessionHldr);
    m_emsSession = emsSessionHldr.value;
}
} catch (Exception ex) {
    // System.out.println("Could not narrow");
ex.printStackTrace();
}

C.2.2.6 Retrieve List of Managers

    manangerNames_THolder names = new manangerNames_THolder();
m_emsSession.getSupportedManagers(names);
managers = names.value;
for (i = 0; i < managers.length; i++) {
    System.out.print("Manager ");
    System.out.print(i);
    System.out.println(" " + managers[i]);
}

C.2.2.7 getEMS Operation on EMS Manager

    EMS_T m_ems;
    EMS_THolder m_emsHolder = new EMS_THolder();
    try {
        Common_IHolder mgrHolder = new Common_IHolder();
m_emsSession.getManager("EMS", mgrHolder);
    EMSSmgr_I emsMgr = EMSSmgr_IHelper.narrow(mgrHolder.value);
    emsMgr.getEMS(m_emsHolder);
    } catch (ProcessingFailureException pfe) {
        System.out.println("Processing Exception" + pfe.getMessage());
pfe.printStackTrace();
    }
m_ems = m_emsHolder.value;
System.out.println("Native EMS Name" + m_ems.nativeEMSName);

C.2.2.8 Get Reference to EventChannel

    EventChannel notifChannel;
    EventChannelHolder chanHolder = new EventChannelHolder();
    try {
    ...    emsSession.getEventChannel(chanHolder);
    } catch (Exception ex) {
        // handle exceptions
    }
    notifChannel = chanHolder.value;
C.2.2.9 Obtain ConsumerAdmin Reference

```
//retrieve default consumer admin
try {
    ConsumerAdmin cadmin = notifChannel.get_consumeradmin(0);
} catch (AdminNotFound anfSe) {
    // Exception handling
}
```

C.2.2.10 Obtain ProxyPushSupplier

```
IntHolder id = new IntHolder();
try {
    ProxySupplier baseSupplier =
        cadmin.obtain_notification_push_supplier(
            ClientType.STRUCTURED_EVENT, id);
    structuredProxyPushSupplier =
        StructuredProxyPushSupplierHelper.narrow(baseSupplier);
} catch (AdminLimitExceeded aleEx) {
    // Exception handling
}
```

C.2.2.11 Implement StructuredPushConsumer

```
class StructuredPushConsumerImpl extends _StructuredPushConsumerPOA {
    StructuredPushConsumerImpl() {
        super();
        System.out.println("StructuredPushConsumerImpl created.");
    }
    public void disconnect_structured_push_consumer() {
        System.out.println("Disconnect structured push consumer.");
    }
    public void push_structured_event(StructuredEvent notification) {
        System.out.println("Received notification.");
    }
    public void offer_change(EventType[] added, EventType[] removed)
        throws InvalidEventType {
        System.out.println("Offer changed.");
    }
}
```
Appendix C

Appendix C: OSS Use Cases and Client Development

C.2.3 Running the Client

If the initial naming context is not resolved using the Option 2 method shown in C.2.2.2 Get Reference to the Naming Service, page C-7, the following JVM flags must be used for CORBA clients written for Orbix 6.2, Visibroker, or jacORB:

- **Orbix 6.2 or later:**
  - `-ORBInitRef NameService=corbaloc:iiop:1.2@<host name>:<port>/NameServiceGWC`

- **Visibroker:**
  - `-DORBInitRef NameService=corbaloc::<host name>:<port>/NameServiceGWC`
  - `-DORBInitRef NameService=corbaname::<host name>:<port>/NameServiceGWC`

- **jacORB:**
  - `-DORBInitRef.NameService=corbaname::<host name>:<port>/NameServiceGWC`

In addition, if you do not set the system properties `org.omg.CORBA.ORBClass` and `org.omg.CORBA.ORBSingletonClass` in your client code before Step 1 in C.2.2.1 Initialize the Client Connection, page C-5, you must set them in your client launch file:

- **Orbix 6.2 or later:**
  - `-Dorg.omg.CORBA.ORBClass=com.iona.corba.art.artimpl.ORBImpl`
  - `-Dorg.omg.CORBA.ORBSingletonClass=com.iona.corba.art.artimpl.ORBSingleton`

- **Visibroker:**
  - `-Dorg.omg.CORBA.ORBClass=com.inprise.vbroker.orb.ORB`
  - `-Dorg.omg.CORBA.ORBSingletonClass=com.inprise.vbroker.orb.ORBSingleton`

- **jacORB:**
  - `-Dorg.omg.CORBA.ORBClass=org.jacorb.orb.ORB`
  - `-Dorg.omg.CORBA.ORBSingletonClass=org.jacorb.orb.ORBSingleton`

For other ORBs, check the related ORB programming guide.
C.2.4 Known Interoperability Issues When Using Orbix

If you use Orbix to set the following property in the jacob.properties file:

`Jacorb.compactTypecodes=0`

Pay attention to the compatibility of the domain you use with Orbix. If you use an insecure domain, set the following property in the jacob.properties file:

`Jacorb.security.support_ssl = off`