IBM® Tivoli® Netcool/OMNIbus Probe for
Nokia-Siemens Switch/Radio/@vantage
Commander (CORBA) 3GPP V6.3
2.0

Reference Guide
November 30, 2012

IBM

Notice

Before using this information and the product it supports, read the information in Appendix A, "Notices and Trademarks," on page 23.

Contents

About this guide

The following sections contain important information about using this guide.

Document control page

Use this information to track changes between versions of this guide.

The IBM Tivoli Netcool/OMNIbus Probe for Nokia-Siemens Switch/Radio/@vantage Commander (CORBA) 3GPP V6.3 documentation is provided in softcopy format only. To obtain the most recent version, visit the IBM® Tivoli® Information Center:

https://www.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/probes/common/Probes.html

Table 1. Document modification history		
Document version	**Publication date**	**Comments**
SC22-5491-00	November 30, 2012	First IBM publication.

Note : The Probe for Nokia-Siemens Switch/Radio/@vantage Commander (CORBA) 3GPP V6.3 (`nco_p_siemens_corba_v2`) was previously documented in the same reference guide as the Probe for Nokia-Siemens Switch/Radio/@vantage Commander (CORBA) 3GPP V5.5 (`nco_p_siemens_corba_v1`). This reference guide documents the Probe for Nokia-Siemens Switch/Radio/@vantage Commander (CORBA) 3GPP V6.3 only.

Conventions used in this guide

All probe guides use standard conventions for operating system-dependent environment variables and directory paths.

Operating system-dependent variables and paths

All probe guides use standard conventions for specifying environment variables and describing directory paths, depending on what operating systems the probe is supported on.

For probes supported on UNIX and Linux operating systems, probe guides use the standard UNIX conventions such as **$**variable for environment variables and forward slashes (**/**) in directory paths. For example:

`$OMNIHOME/probes`

For probes supported only on Windows operating systems, probe guides use the standard Windows conventions such as **%**variable**%** for environment variables and backward slashes (****) in directory paths. For example:

`%OMNIHOME%\probes`

For probes supported on UNIX, Linux, and Windows operating systems, probe guides use the standard UNIX conventions for specifying environment variables and describing directory paths. When using the Windows command line with these probes, replace the UNIX conventions used in the guide with Windows conventions. If you are using the bash shell on a Windows system, you can use the UNIX conventions.

Note : The names of environment variables are not always the same in Windows and UNIX environments. For example, %TEMP% in Windows environments is equivalent to $TMPDIR in UNIX and Linux environments. Where such variables are described in the guide, both the UNIX and Windows conventions will be used.

Operating system-specific directory names

Where Tivoli Netcool/OMNIbus files are identified as located within an *arch* directory under NCHOME or OMNIHOME, *arch* is a variable that represents your operating system directory. For example:

`$OMNIHOME/probes/arch`

The following table lists the directory names used for each operating system.

Note : This probe may not support all of the operating systems specified in the table.

Table 2. Directory names for the arch variable	
Operating system	**Directory name represented by *arch***
AIX® systems	`aix5`
Red Hat Linux® and SUSE systems	`linux2x86`
Linux for System z	`linux2s390`
Solaris systems	`solaris2`
Windows systems	`win32`

OMNIHOME location

Probes and older versions of Tivoli Netcool/OMNIbus use the OMNIHOME environment variable in many configuration files. Set the value of OMNIHOME as follows:

- On UNIX and Linux, set `$OMNIHOME` to `$NCHOME/omnibus`.
- On Windows, set `%OMNIHOME%` to `%NCHOME%\omnibus`.

Chapter 1. Probe for Nokia-Siemens Switch/Radio/@vantage Commander (CORBA) 3GPP V6.3

The IBM Tivoli Netcool/OMNIbus Probe for Nokia-Siemens Switch/Radio/@vantage Commander (CORBA) 3GPP V6.3 collects alarms from the Nokia-Siemens Switch Commander, Radio Commander, and @vantage Commander element management systems (EMS).

The probe collects alarms from the Nokia-Siemens EMS using the Common Object Request Broker Architecture (CORBA) 3rd Generation Partnership Project (3GPP) V6.3 interface.

The probe complies with the following 3GPP standards:

Table 3. Supported 3GPP standards		
3GPP Technical Specification	**Version**	**Object**
32.111-3	6.3.0	Alarm Integration Reference Point (IRP)
32.303	6.4.0	Notification IRP
32.363	6.3.0	Entry Point IRP

This guide contains the following sections:

Summary

Each probe works in a different way to acquire event data from its source, and therefore has specific features, default values, and changeable properties. Use this summary information to learn about this probe.

The following table provides a summary of the probe.

Table 4. Summary	
Probe target	Nokia-Siemens CORBA 3GPP V6.3 interface
Probe executable name	`nco_p_siemens_corba_v2`
Probe installation package	`omnibus-`*`arch`*`-probe-nco-p-siemens-corba-v2-`*`version`*
Package version	2.0

Table 4. Summary (continued)	
Probe supported on	For details of supported operating systems, see the following Release Notice on the IBM Software Support website: https://www-304.ibm.com/support/docview.wss?uid=swg21613417
Properties file	`$OMNIHOME/probes/`*arch*`/siemens_corba_v2.props`
Rules file	`$OMNIHOME/probes/`*arch*`/siemens_corba_v2.rules`
Requirements	For details of any additional software that this probe requires, refer to the `description.txt` file that is supplied in its download package.
Connection method	CORBA
Multicultural support	Not Available
Peer-to-peer failover functionality	Available
IP environment	IPv4 and IPv6
Federal Information Processing Standards (FIPS)	IBM Tivoli Netcool/OMNIbus uses the FIPS 140-2 approved cryptographic provider: IBM Crypto for C (ICC) certificate 384 for cryptography. This certificate is listed on the NIST website at http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2004.htm. For details about configuring Netcool/OMNIbus for FIPS 140-2 mode, see the *IBM Tivoli Netcool/OMNIbus Installation and Deployment Guide*.

Installing probes

All probes are installed in a similar way. The process involves downloading the appropriate installation package for your operating system, installing the appropriate files for the version of Netcool/OMNIbus that you are running, and configuring the probe to suit your environment.

The installation process consists of the following steps:

1. Downloading the installation package for the probe from the Passport Advantage Online website.

   Each probe has a single installation package for each operating system supported. For details about how to locate and download the installation package for your operating system, visit the following page on the IBM Tivoli Knowledge Center:

   http://www-01.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/probes/all_probes/wip/reference/install_download_intro.html

2. Installing the probe using the installation package.

   The installation package contains the appropriate files for all supported versions of Netcool/OMNIbus. For details about how to install the probe to run with your version of Netcool/OMNIbus, visit the following page on the IBM Tivoli Knowledge Center:

   http://www-01.ibm.com/support/knowledgecenter/SSSHTQ/omnibus/probes/all_probes/wip/reference/install_install_intro.html

3. Configuring the probe.

This guide contains details of the essential configuration required to run this probe. It combines topics that are common to all probes and topics that are peculiar to this probe. For details about additional configuration that is common to all probes, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

Firewall considerations

When using CORBA probes in conjunction with a firewall, the firewall must be configured so that the probe can connect to the target system.

Most CORBA probes can act as both a server (listening for connections from the target system) and a client (connecting to the port on the target system to which the system writes events). If you are using the probe in conjunction with a firewall, you must add the appropriate firewall rules to enable this dual behavior.

There are three possible firewall protection scenarios, for which you must determine port numbers before adding firewall rules:

1. If the host on which the probe is running is behind a firewall, you must determine what remote host and port number the probe will connect to.

2. If the host on which the target system is running is behind a firewall, you must determine the incoming port on which the probe will listen and to which the target system will connect.

3. If each host is secured with its own firewall, you must determine the following four ports:

   a. The outgoing port (or port range) for the probe.

   b. The hostname and port of the target system.

   c. The outgoing port on which the target system sends events if the probe is running as a client.

   d. The incoming port on which the probe listens for incoming events.

Note : Most, but not all, CORBA probes listen on the port specified by the **ORBLocalPort** property. The default value for this property is 0, which means that an available port is selected at random. If the probe is behind a firewall, the value of the **ORBLocalPort** property must be specified as a fixed port number.

CORBA probes that use EventManager or NotificationManager objects may use different hosts and ports from those that use NamingService and EntryPoint objects. If the probe is configured to get object references from a NamingService or EntryPoint object, you must obtain the host and port information from the system administrator of the target system. When you have this information, you can add the appropriate firewall rules.

Data acquisition

Each probe uses a different method to acquire data. Which method the probe uses depends on the target system from which it receives data.

The probe gathers alarms from the Nokia-Siemens EMS using the CORBA 3GPP interface.

Data acquisition is described in the following topics:

- "Connecting to the CORBA interface" on page 4
- "Retrieving objects" on page 4
- "Filtering notifications and alarms" on page 4
- "Populating tokens during resynchronization" on page 6
- "Lookup table" on page 6
- "IRP Agent status" on page 7
- "Command line interface" on page 7
- "Peer-to-peer failover functionality" on page 8

Connecting to the CORBA interface

The probe acts as an Integration Reference Point (IRP) Manager and connects to the Nokia-Siemens EMS using a CORBA 3GPP interface.

The `AlarmIRPOperation` and `NotificationIRPOperation` points form a part of the IRP Agent to which the probe connects.

Depending on your operating environment, use one of the following connection configurations:

1. Option 1 - specify values for the following properties:
   - **AlarmIrpFile** - specify the name used to resolve the Alarm IRP in the CORBA Naming Service.
   - **NotificationIrpFile** - specify the name used to resolve the Notification IRP in the CORBA Naming Service.

2. Option 2 - specify values for the following properties:
   - **NamingServiceHost** - specify the name of the host machine where the CORBA Naming Service is running.
   - **NamingServicePort** - specify the port on which the CORBA Naming Service listens.
   - **AlarmIrpName** - specify the name used to resolve the Alarm IRP in the CORBA Naming Service.
   - **NotificationIrpName** - specify the name used to resolve the Notification IRP in the CORBA Naming Service.

3. Option 3 - specify values for the following properties:
   - **EntryPointIrpFile** - specify the location of the file that contains the Interoperable Object Reference (IOR) of the EntryPoint IRP object.
   - **NotificationIrpName** - specify the name used to resolve the Notification IRP in the CORBA Naming Service.
   - **AlarmIrpName** - specify the name used to resolve the Alarm IRP in the CORBA Naming Service.

Retrieving objects

If the **Resynch** property is set to `true`, the probe initially receives a list of all active alarms from the `AlarmIRP` point.

The probe then connects to the `NotificationIRP` point and uses the CORBA notification push model to receive new alarms. If the **Resynch** property is set to `false`, the probe only receives new alarms.

Filtering notifications and alarms

You can specify the notifications and alarms that the probe receives from the target device using the **NotificationFilter** and **AlarmFilter** properties.

When you use these properties, you must use the actual token names. For example, the token h represents the element NV_PERCEIVED_SEVERITY. So to specify that the probe is only sent notifications with a perceived severity of 3, set the **NotificationFilter** property to $h == 3.

You can specify more complex filters using AND, OR, and NOT statements. For example, to specify that the probe is sent notifications with a perceived severity of 3 or 4, set the **NotificationFilter** property to the following value:

($h == 3) OR ($h == 4)

Specifying specific managed elements

To ensure that the probe is only sent notifications for a specific managed element, specify a value of *Managed_Node_Name~$f* for the **NotificationFilter** property, where *Managed_Node_Name* is the name of the managed object and *$f* represents the element NV_MANAGED_OBJECT_INSTANCE.

For example, if you require a set of alarms that return a value of SubNetwork=ONRM_RootMo, SubNetwork=SNMP, ManagedElement=SP1 for the NV_MANAGED_OBJECT_INSTANCE element , specify the following value for the **NotificationFilter** property:

SP1'~$'f

Note : When specifying values for the **NotificationFilter** property, you must replace spaces with the tilde character (~). For string comparisons, the first argument is considered to be contained in the second argument, which is why *$f* is positioned after the literal.

Token/element mappings

The following table lists the token/element mappings available for the **AlarmFilter** and **NotificationFilter** properties:

Table 5. Token/element mappings	
Token	**Element**
a	NV_NOTIFICATION_ID
b	NV_EVENT_TIME
c	NV_SYSTEM_DN
d	NV_MANAGED_OBJECT_CLASS
e	NV_MANAGED_OBJECT_INSTANCE
f	NV_ALARM_ID
ff	NV_ALARM_LIST_ALIGNMENT_REQUIREMENT
g	NV_PROBABLE_CAUSE
gg	NV_SERVICE_USER
h	NV_PERCEIVED_SEVERITY
hh	NV_SERVICE_PROVIDER
i	NV_SPECIFIC_PROBLEM
ii	NV_SECURITY_ALARM_DETECTOR
j	NV_ADDITIONAL_TEXT
jj	NV_VENDOR_SPECIFIC_ALARM_TYPE
k	NV_ACK_TIME
l	NV_ACK_USER_ID
m	NV_ACK_SYSTEM_ID
n	NV_ACK_STATE

Table 5. Token/element mappings (continued)	
Token	**Element**
o	NV_COMMENTS
p	NV_BACKED_UP_STATUS
q	NV_BACK_UP_OBJECT
r	NV_THRESHOLD_INFO
s	NV_TREND_INDICATION
t	NV_STATE_CHANGE_DEFINITION
u	NV_MONITORED_ATTRIBUTES
v	NV_PROPOSED_REPAIR_ACTIONS
w	NV_CORRELATED_NOTIFICATIONS
x	NV_REASON
y	CLEAR_USER_ID
z	CLEAR_SYSTEM_ID

Populating tokens during resynchronization

The `StructuredEvent` structure from the `CosNotifications.idl` interface description language (IDL) file contains a `remainder_of_body` field that has the following structure definition:

```
struct StructuredEvent { EventHeader header;
FilterableEventBody filterable_data; any remainder_of_body;
}; // StructuredEvent
```

The `remainder_of_body` field is only present in notifications, where it can be used for storing nonfilterable fields such as NV_ALARM_ID. The probe will not populate tokens other than the standard filterable ones for alarms received during resynchronization. For example, the probe cannot populate the NV_ALARM_ID field for alarms received during resynchronization.

Lookup table

The probe is supplied with a lookup table that contains details of the various types of alarms that the Nokia-Siemens EMS generates.

At installation, the `Corba_3gpp_V630.lookup` file supplied with the probe installation package is installed to the following location:

`$OMNIHOME/probes/includes/`

The following line in the rules file references the lookup file:

`include "../includes/Corba_3gpp_V630.lookup"`

Note : The include command assumes that the probe is run from the standard `$OMNIHOME/probes/` directory. If you are running the probe from a different directory, replace ".." with the absolute directory path to the lookup table. Do not use the $OMNIHOME environment variable in this directory path.

IRP Agent status

The probe checks the status of the IRP Agent every 60 seconds.

You can use the **Agentheartbeat** property to change the frequency with which the probe checks the status of the IRP Agent.

Command line interface

The probe is supplied with a command line interface (CLI). This interface enables you to execute commands to acknowledge alarms or request a full resynchronization in the CORBA interface.

To use the CLI, you must use the **CommandPort** property to specify a port through which commands will be sent. The default port is 6970. When you want to issue commands, use Telnet to connect through this port. You can use the **CommandPortLimit** property to limit the number of Telnet connections that the probe can make at one time.

The following table describes the commands that you can use with the CLI:

Table 6. CLI commands	
Command	**Description**
acknowledge_alarm *alarm_id*	Use this command to acknowledge an alarm in the CORBA interface by specifying the identifier (NV_ALARM_ID) of the alarm. Before you can use this command, you must specify a value for the **AckSystemId** property.
userid_acknowledge_alarm *alarm_id* *user_id*	Use this command to acknowledge an alarm in the CORBA interface by specifying the identifier (NV_ALARM_ID) of the alarm and the user identifier (NV_ACK_USER_ID) of the user acknowledging the alarm. Before you can use this command, you must specify a value for the **AckSystemId** property. If you specify a value for the **AckUserId** property, the *user_id* parameter is not required.
unacknowledge_alarm *alarm_id*	Use this command to unacknowledge an alarm in the CORBA interface by specifying the identifier (NV_ALARM_ID) of the alarm. Before you can use this command, you must specify a value for the **AckSystemId** property.
userid_unacknowledge_alarm *alarm_id* *user_id*	Use this command to unacknowledge an alarm in the CORBA interface by specifying the identifier (NV_ALARM_ID) of the alarm and the user identifier (NV_ACK_USER_ID) of the user acknowledging the alarm. Before you can use this command, you must specify a value for the **AckSystemId** property. If you specify a value for the **AckUserId** property, the *user_id* parameter is not required.

Table 6. CLI commands (continued)	
Command	**Description**
clear_alarm *alarm_id*	Use this command to clear an alarm by specifying the identifier (NV_ALARM_ID) of the alarm. Before you can use this command, you must specify a value for the **ClearSystemId** property.
userid_clear_alarm *alarm_id user_id*	Use this command to clear an alarm by specifying the identifier (NV_ALARM_ID) of the alarm and the user identifier (NV_ACK_USER_ID) of the user clearing the alarm. Before you can use this command, you must specify a value for the **ClearSystemId** property. If you specify a value for the **ClearUserId** property, the *user_id* parameter is not required.
resynch_all	Use this command to perform a full resynchronization with the CORBA interface.
resynch_filter *filter*	Use this command to perform a partial resynchronization with the CORBA interface. The *filter* parameter uses the same filter format as the **AlarmFilter** property.
help	This command displays online help about the CLI.
exit/quit	This command closes the connection.
version	This command displays the version of the probe.

CLI scripts

Because the CLI uses Telnet connections, you can connect to the probe from anywhere by creating a desktop tool to open a Telnet connection, send a command, and then close the connection. This means that simple scripts can be set up to allow users to acknowledge selected events from the IBM Tivoli Netcool/OMNIbus event list.

Peer-to-peer failover functionality

The probe supports failover configurations where two probes run simultaneously. One probe acts as the master probe, sending events to the ObjectServer; the other acts as the slave probe on standby. If the master probe fails, the slave probe activates.

While the slave probe receives heartbeats from the master probe, it does not forward events to the ObjectServer. If the master probe shuts down, the slave probe stops receiving heartbeats from the master and any events it receives thereafter are forwarded to the ObjectServer on behalf of the master probe. When the master probe is running again, the slave probe continues to receive events, but no longer sends them to the ObjectServer.

Example property file settings for peer-to-peer failover

You set the peer-to-peer failover mode in the properties files of the master and slave probes. The settings differ for a master probe and slave probe.

Note : In the examples, make sure to use the full path for the property value. In other words replace $OMNIHOME with the full path. For example: /opt/IBM/tivoli/netcool.

The following example shows the peer-to-peer settings from the properties file of a master probe:

```
Server     :     "NCOMS"
RulesFile  :     "master_rules_file"
MessageLog :     "master_log_file"
PeerHost   :     "slave_hostname"
PeerPort   :     6789 # [communication port between master and slave probe]
Mode       :     "master"
PidFile    : "master_pid_file"
```

The following example shows the peer-to-peer settings from the properties file of the corresponding slave probe:

```
Server     :     "NCOMS"
RulesFile  :     "slave_rules_file"
MessageLog :     "slave_log_file"
PeerHost   :     "master_hostname"
PeerPort   :     6789 # [communication port between master and slave probe]
Mode       :     "slave"
PidFile    : "slave_pid_file"
```

Properties and command line options

You use properties to specify how the probe interacts with the device. You can override the default values by using the properties file or the command line options.

The following table describes the properties and command line options specific to this probe. For more information about generic Netcool/OMNIbus properties and command line options, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

Table 7. Properties and command line options		
Property Name	**Command Line Option**	**Description**
AckSystemId *string*	-acksystemid *string*	Use this property to specify the system identifier to use when acknowledging and deacknowledging alarms using the CLI. The default is " ".
AckUserId *string*	-ackuserid *string*	Use this property to specify the user ID to use when acknowledging and deacknowledging alarms using the CLI. The default is " ".
Agentheartbeat *integer*	-agentheartbeat *integer*	Use this property to specify the frequency (in seconds) with which the probe checks the status of the IRP Agent. The default is 60.
AlarmFilter *string*	-alarmfilter *string*	Use this property to specify the filter that the Alarm IRP point uses to limit the alarms sent to the probe. The default is " ".

Table 7. Properties and command line options (continued)

Property Name	Command Line Option	Description
AlarmIrpFile *string*	`-alarmirpfile` *string*	Use this property to specify the location of the file that contains the IOR of the AlarmIRP object. The default is "".
AlarmIrpName *string*	`-alarmirpiorfilename` *string*	Use this property to specify the name of the Alarm IRP Agent. The default is `1.SubNetwork/ 1.Management Node/1.IrpAgent/ A0.AlarmIRP/Object`.
BeatInterval *integer*	`-beatinterval` *integer*	Use this property to specify the heartbeat interval (in seconds) for peer-to-peer failover. The default is 2.
ClearSystemId *string*	`-clearsystemid` *string*	Use this property to specify the system identifier to use when clearing alarms using the CLI. The default is "".
ClearUserId *string*	`-clearuserid` *string*	Use this property to specify the user ID to use when clearing alarms using the CLI. The default is "".
CommandPort *integer*	`-commandport` *string*	Use this property to specify the port through which you will send commands using the CLI. The default is 6970.
CommandPortLimit *integer*	`-commandportlimit` *integer*	Use this property to specify the maximum number of Telnet connections that the probe can make using the command port. The default is 10.
EntryPointIrpFile *string*	`-entrypointirpfile` *string*	Use this property to specify the location of the file containing the IOR of the Entry Point IRP object. The default is "".
FlushBufferInterval *integer*	`-flushbufferinterval` *integer*	Use this property to specify how often (in seconds) the probe flushes all alerts in the buffer to the ObjectServer. The default is 0 (which instructs the probe to never flush the alerts to the ObjectServer).

Table 7. Properties and command line options (continued)		
Property Name	**Command Line Option**	**Description**
Mode *string*	`-master` (This is equivalent to **Mode** with a value of `master`.) `-slave` (This is equivalent to **Mode** with a value of `slave`.)	Use this property to specify the mode of operation of a probe instance in a peer-to-peer failover relationship. This property takes the following values: `master`: The probe is the master probe in the failover relationship. `slave`: The probe is the slave probe in the failover relationship. `standard`: There is no failover relationship. The default is `standard`.
NamingServiceHost *string*	`-namingservicehost` *string*	Use this property to specify the name of the host where the Naming Service is available. The default is `localhost`.
NamingServicePort *string*	`-namingserviceport` *string*	Use this property to specify the port on which the Naming Service listens on. The default is `5001`.
NonFilterableEvent BodyResynchParse *string*	`-nonfilterableevent bodyresynchparse` (This is equivalent to **NonFilterableEvent BodyResynchParse** with a value of `true`.) `-nononfilterableevent bodyresynchparse` (This is equivalent to **NonFilterableEvent BodyResynchParse** with a value of `false`.)	Use this property to specify whether the probe parses the `NonFilterableEventBody` structure during resynchronization. The default is `true`.
NotificationCategori es *string*	`-notificationcategories` *string*	Use this property to specify the notification categories to which the probe subscribes. To specify multiple categories, separate them using semicolons in the following format: `category1;category2;category`*n* The default is `""` (the probe subscribes to all available notification categories).
NotificationFilter *string*	`-notificationfilter` *string*	Use this property to specify the filter that the Notification IRP point uses to limit the notifications sent to the probe. The default is `""`.

Table 7. Properties and command line options (continued)

Property Name	Command Line Option	Description
NotificationIrpFile *string*	`-notificationirpfile` *string*	Use this property to specify the location of the file that contains the IOR of the `NotificationIRP` object. The default is `""`.
NotificationIrpName *string*	`-notificationirpname` *string*	Use this property to specify the name of the Notification IRP Agent. The default is `1.SubNetwork/ 1.Management Node/1.IrpAgent/ N0.NotificationIRP/Object`.
NotificationIrpVersi on *string*	`- notificationirpversion` *string*	Use this property to specify the Notification IRP version. The version specified must be use the format `\"32.303 VN.n\"`, where N is the major version number and n is the minor version number. For example: 32.303 V6.4 The default is `""`.
ORBCharEncoding *string*	`-orbcharencoding` *string*	Use this property to specify the native character encoding set used by the Object Request Broker (ORB) for character data. The default is `ISO8859_1`.
ORBLocalPort *integer*	`-orblocalport` *integer*	Use this property to specify the local port to which the ORB listens. The default is `0` (the ORB selects an available port at random).
ORBLocalHostName *string*	`-orblocalhostname` *string*	Use this property to specify the local host used by the server-side ORB to place the server's host name into the IOR of a remote object. The default is `""`.
ORBWCharDefault *string*	`-orbwchardefault` *string*	Use this property to specify what wide character (wchar) set the IBM ORB uses when communicating with other ORBs that do not publish a wchar set. The default is `UTF16`.
PeerHost *string*	`-peerhost` *string*	Use this property to specify the host name of the network element acting as the counterpart to this probe instance in a peer-to-peer failover relationship. The default is `localhost`.

Table 7. Properties and command line options (continued)

Property Name	Command Line Option	Description
PeerPort *integer*	`-peerport` *integer*	Use this property to specify the port through which the master and slave probes communicate in a peer-to-peer failover relationship. The default is 9999.
Resynch *string*	`-noresynch` (This is equivalent to **Resynch** with a value of `false`.) `-resynch` (This is equivalent to **Resynch** with a value of true.)	Use this property to specify whether the probe attempts to resynchronize existing alarms before retrieving new alarms. This property takes the following values: `false` : The probe does not attempt to resynchronize the alarms. `true` : The probe attempts to resynchronize the alarms. The default is `false`.
Retry *string*	`-noretry` (This is equivalent to **Retry** with a value of `false`.) `-retry` (This is equivalent to **Retry** with a value of `true`.)	Use this property to specify whether or not the probe attempts to reconnect to the CORBA interface following a time out. This property takes the following values: `false` : The probe does not attempt to reconnect to the CORBA interface. `true` : The probe attempts to reconnect to the CORBA interface. The default is `false`.
Timeout *integer*	`-timeout` *integer*	Use this property to specify the length of time (in seconds) that the probe waits to receive events before disconnecting from the Notification Service and shutting down. The default is 0 (the probe never times out).
TimeTick *integer*	`-timetick` *integer*	Use this property to specify the duration (in minutes) of the subscription to the target. The default is 0.

Elements

The probe breaks event data down into tokens and parses them into elements. Elements are used to assign values to ObjectServer fields. The field values contain the event details in a form that the ObjectServer understands.

The following table describes the elements that the probe generates.

Table 8. Elements	
Element name	**Element description**
CLEAR_SYSTEM_ID	This element identifies the system where the alarms in the IRP Manager are cleared.
CLEAR_USER_ID	This element contains the name of the user who cleared an alarm.
DOMAIN_NAME	This element contains the domain name from which a notification originated.
EVENT_NAME	This element indicates the notification type of a notification.
EVENT_TYPE	This element indicates the event type of a notification.
NV_ACK_STATE	This element indicates the acknowledgement state of an alarm.
NV_ACK_SYSTEM_ID	This element contains the System ID of the IRP Manager processing a notification.
NV_ACK_TIME	This element indicates the time at which the user acknowledged an alarm.
NV_ACK_USER_ID	This element identifies the last user who has changed the acknowledgement state.
NV_ADDITIONAL_TEXT	This element contains information about the network element from which an alarm originated.
NV_ALARM_ID	This element contains the identifier of an alarm as it appears in the alarm list.
NV_ALARM_LIST_ALIGNMENT_REQUIREME NT	This element indicates whether or not the alarm list requires alignment.
NV_BACK_UP_OBJECT	This element contains the Distinguished Name (DN) of the backup object.
NV_BACKED_UP_STATUS	This element indicates whether the object has been backed up.
NV_COMMENTS	This element contains comments about an alarm.
NV_CORRELATED_NOTIFICATIONS	This element contains a set of notifications to which a notification is related. This element is generated dynamically and its content depends on the IRP Agent.
NV_EVENT_TIME	This element indicates the time at which an event occurred.

Table 8. Elements (continued)	
Element name	**Element description**
NV_MANAGED_OBJECT_CLASS	This element contains the Information Object Class of the managed network resource.
NV_MANAGED_OBJECT_INSTANCE	This element contains the Managed Object Instance of the network resource.
NV_MONITORED_ATTRIBUTES	This element contains the Managed Object attributes of the network resource whose value changes are being monitored.
NV_NOTIFICATION_ID	This element contains the identification information of a notification.
NV_PERCEIVED_SEVERITY	This element indicates the relative level of urgency of an alarm.
NV_PROBABLE_CAUSE	This element provides further information about the probable cause of an alarm.
NV_PROPOSED_REPAIR_ACTIONS	This element indicates the proposed repair actions associated with a notification.
NV_REASON	This element indicates the reason that triggered the proposed repair action.
NV_SECURITY_ALARM_DETECTOR	This element indicates the security alarm detector for the device.
NV_SERVICE_PROVIDER	This element contains the name of the service provider.
NV_SERVICE_USER	This element contains the name of the service user whose request for service led to the generation of a security alarm.
NV_SPECIFIC_PROBLEM	This element contains further information about the problem to which a notification relates.
NV_STATE_CHANGE_DEFINITION	This element contains information about the state change.
NV_SYSTEM_DN	This element indicates the Distinguished Name (DN) used to identify the system.
NV_THRESHOLD_INFO	This element provides information about a threshold that has been crossed.
NV_TREND_INDICATION	This element indicates how an observed condition has changed.
NV_VENDOR_SPECIFIC_ALARM_TYPE	This element indicates the alarm type specific to the vendor.

Error messages

Error messages provide information about problems that occur while running the probe. You can use the information that they contain to resolve such problems.

The following table describes the error messages specific to this probe. For information about generic Netcool/OMNIbus error messages, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

Table 9. Error messages

Error Message	Description	Action
`[Command Port]` *host* `Failed to get socket IO` *io* `[Command Port]` *host* `Failed to read command` *exception* `[Command Port] Error occurred` *exception* `[Command Port] Failed to get property 'CommandPort'` `[Command Port]` *host* `Failed to close command socket` *exception*	There was a problem with the command port.	Check that you have specified the command port properties correctly. Check the connection between the probe and the command port.
`[Command Port] Failed to get CommandPortLimit property - using 10`	The probe could not retrieve the value of the **CommandPortLimit** property. The probe will use the default value of 10.	Check that you have specified the command port correctly. Check the connection between the probe and the command port.
`[Command Port] Failed to open listening socket, shutting down Thread!`	The probe could not open a listening socket on the command port.	Try using another port.
`[Command Port] Thread shutting down due to error!`	There was a problem with the command port.	Check that the command port has been specified correctly and that the `nco_p_nonnative` process is running.
`Cannot initialize the Orb` *exception*	A problem was encountered during initialization of the Object Request Broker (ORB).	Ensure the CLASSPATH environment variable contains the path to the ORB `.jar` files.
`Cannot proceed. Shutting down!`	There is a problem with your network or with the probe configuration.	Contact IBM Software Support.

Table 9. Error messages (continued)

Error Message	Description	Action

`Communication failure - lost connection to NoticiationIRP:` *exception*
 	
`CORBA.TRANSIENT exception raised. NotificationIRP is down!`
 	
`CORBA.OBJECT_NOT_EXIST exception raised. NotificationIRP is down!`
 	
`BAD_PARAM Exception i.e one or more of the in/out parameter is null` *exception*
 	
`CORBA Exception stack trace to stderr` *exception*
 	
`NetcoolIRPManager: Stack Trace to stderr` *exception*
	Communication with the Notification IRP server has failed.	Check that the Notification IRP server is running.

`Failed to Connect` *exception*	The server is not running, the IOR is out of date, or the probe cannot reach the remote server.	Check that the relevant properties are set correctly and that the target host is working correctly.
`Failed to connect to the Naming Context` *exception*	The probe cannot connect to the Naming Context.	Check the values of the **NamingServiceHost** and **NamingServicePort** properties.
`Failed to iterate through resynch alarms`	A problem occurred while the probe was parsing the alarms retrieved during a resynchronization.	Check that the server is running correctly. Check that you have specified the resynchronization parameters correctly.
`Failed to perform resynch`	The probe failed to get the alarm list or failed to iterate through the resynchronization alarms.	Check the value specified for the **Resynch** property.
`Failed to ping notification service`	The probe encountered connection problems with the Notification IRP point.	Check the value specified for the **Agentheartbeat** property.
`Failed to resolve the AlarmIRP object` *exception*	The Alarm IRP object is not registered in the Naming Service with the Alarm IRP name provided in the properties file.	Check the value specified for the **AlarmIrpName** property.
`Failed to resolve the NotificationIRP object` *exception*	The Notification IRP object is not registered in the Naming Service with the Notification IRP Name provided in the properties file.	Check the value specified for the **NotificationIrpName** property.

Table 9. Error messages (continued)

Error Message	Description	Action
`login: Unknown exception occurred` *`exception`*	Problems getting the IRP Object.	Check the values specified for the **`AlarmIrpName`**, **`NotificationIrpName`**, and **`EntryPointIrpFile`** properties.
`NetcoolIRPManager: ERROR when parsing a notification event`	The probe encountered corrupted data while parsing. The data is not in the expected format.	Check the settings in the notification and problem log files.
`NetcoolIRPManager: Failed to acknowledge_alarms():` *`exception`*	The probe could not acknowledge the alarm.	Check that the alarm identifier provided to the CLI is correct.
`NetcoolIRPManager: Failed to retrieve AlarmIRP object from security interface`	Failed to get the Alarm IRP object.	Check the value specified for the **`AlarmIrpName`** property.
`NetcoolIRPManager: Failed to retrieve NotificationIRP object from security interface`	Failed to get the Notification IRP object.	Check the value specified for the **`NotificationIrpName`** property.
`NetcoolIRPManager: Failed to send event` *`exception`*	The probe could not forward the event to the ObjectServer.	Check that the ObjectServer is running.
`NetcoolIRPManager: Failed to Unacknowledge_alarms():` *`exception`*	The probe could not unacknowledge the alarm.	Check the connection between the probe and the command port. Check that you have specified the command port correctly.
`OperationNotSupported` *`exception`* `InvalidParameter Exception:` *`exception`*	The Nokia-Siemens EMS does not support the operation specified.	Check the value specified for the **`NotificationIrpName`** property.
`Problem while trying to connect to the IRP points` *`exception`*	A problem occurred while connecting to the Alarm IRP or the Notification IRP.	Check the values specified for the**`AlarmIrpName`** and **`NotificationIrpName`** properties.
`Login: Failed to narrow Entry Point IRP interface:` *`exception`*	The probe failed to narrow the object reference to the Entry Point IRP Agent object.	Check the value of the **`EntryPointIRPFile`** property in the properties file.

Error Message	Description	Action
Table 9. Error messages (continued)		
Error Message	**Description**	**Action**

`Failed to retrieve property AlarmIrpFile`
 	
`Failed to retrieve property NotificationIrpFile`
	The probe could not find one or more properties required to connect to the server.	Check that the properties given in the message are present in the properties file and that they have the correct values.

`Failed to find IOR file 'alarmirp_ior_file'`	The probe has failed to find the IOR file.	Check that the **AlarmIrpFile** property is set correctly in the properties file.

`Failed to find IOR file`
 	
`Login: Failed to find IOR file 'ior_file'`
	The probe has failed to find the IOR file.	Check that the **EntryPointIrpFile** property is set correctly in the properties file.

`Failed to get subscription status: exception`	The probe cannot get the subscription status from the notification service.	Check that the server is running correctly. Check the connection to the server.
`VIV:printStackTrace:`		
`Failed to Connect to notification service: exception`	The probe lost its connection to the server.	Check that the server is running correctly. Check the connection to the server.
`Either, the server is not running, the IOR is out of date, or probe cannot reach remote Server!`	There are no responses from the Notification IRP point.	Check that the server is running correctly. Check the connection to the server. Check that the IOR is not out of date. Check that the probe has the permissions required to log into the server.
`NetcoolIRPManager: Failed to clear_alarms: exception`	The probe has failed to clear the alarms.	Check that the NV_ALARM_ID specified in the CLI is correct. Check that the port number is correct. Check the connection to the server.

`Error. Failed to parse event completely`
 	
`Name is null, cannot create Element`
	The probe failed to parse the alarm.	Check that the alarm conforms to the `StructuredEvent` format. Check that the server is running correctly.

`Will listen for commands on port number listening_port`	An information message identifying the port that the probe will listen on for commands.	None.
`Failed to listen for commands on port number listening_port: io`	The probe was unable to listen for commands on the specified. port.	Check that the **CommandPort** property is set correctly in the properties file.

ProbeWatch messages

During normal operations, the probe generates ProbeWatch messages and sends them to the ObjectServer. These messages tell the ObjectServer how the probe is running.

The following table describes the ProbeWatch messages that the probe generates. For information about generic Netcool/OMNIbus ProbeWatch messages, see the *IBM Tivoli Netcool/OMNIbus Probe and Gateway Guide*.

Table 10. ProbeWatch Messages

ProbeWatch message	Description	Triggers or causes

`Communication failure - lost connection to NoticiationIRP` *exception*
 	
`CORBA.TRANSIENT Exception: Lost connection to NotificationIRP` *exception*
 	
`CORBA.OBJECT_NOT_EXIST Exception: Lost connection to NotificationIRP` *exception*
 	
`Failed to connect to NotificationIRP:` *reason*`. Shutting down.`
 	
`Failed to Connect: Either, the server is not running, the IOR is out of date, or probe cannot reach remote Server`
	The communication with the Notification IRP server has failed.	The probe lost its connection to the Notification IRP server.

`END SYNCHRONIZATION`	The synchronization of alarms has stopped.	The probe has resynchronized all the available alarms.
`Failed to find IOR file '`*alarmirp_ior_file*`'`	The specified Alarm IRP file does not exist.	The IRP Manager has been set up incorrectly.
`Failed to find IOR file '`*notificationirp_ior_file*`'`	The specified Notification IRP file does not exist.	The IRP Manager has been set up incorrectly.
`Failed to get IRP Outline` *exception*	The probe could not get the supported IRP outline using Entry Point IRP.	Incorrect value set for the **AlarmIrpName** or **NotificationIrpName** properties.
`Failed to get IRP reference` *exception* `Failed to get IRP references - invalid parameter`	The probe could not get the IRP references using Entry Point IRP.	Failed to get the IRP Outline.

Table 10. ProbeWatch Messages (continued)

ProbeWatch message	Description	Triggers or causes
`Failed to listen for commands on port number` *`listening_port`* `: io`	A problem occurred while trying to listen for commands.	There was a problem initializing the connection due to insufficient memory or (if this message was sent after some events had been parsed) there was a connection failure.
`Invalid IRPVersion`	The specified IRP version is incorrect.	An incorrect IRP name is specified in the properties file.
`START SYNCHRONIZATION`	The synchronization of alarms has started.	The probe started to resynchronize alarms collected in the system during the timeout period.
`Will listen for commands on port number` *`listening_port`*	The probe will listen for commands on the specified port number.	The probe has successfully created the CLI server on the specified command port specified in the properties file.

`Unexpected fatal error, failed to connect` *`exception`*
 	
`Unexpected fatal error while getting IRP Outline`
 	
`Unexpected fatal error when getting IRP reference from Entry Point IRP`
	The probe has encountered a fatal error.	Contact IBM Software Support.

`Login: Failed to find IOR file '`*`ior_file`*`'`	The specified IOR file does not exist.	The IRP Manager has been set up incorrectly.
`Failed to get subscripton status:` *`exception`*	The probe cannot set subscription information from the notification service.	

Running the probe

Before running the probe for the first time, you must specify a minimum set of properties.

Depending on which method you use to connect to the CORBA 3GPP interface, you must specify values for the some of the following properties before running the probe:

- **AlarmIrpFile**
- **AlarmIrpName**
- **EntryPointIrpFile**
- **NamingServiceHost**
- **NamingServicePort**
- **NotificationIrpFile**

- **NotificationIrpName**

For more information about these properties, see "Connecting to the CORBA interface" on page 4.

Starting the probe

Use the following command to start the probe:

```
$OMNIHOME/probes/nco_p_siemens_corba_v2
```

Shutting down the probe

To stop an instance of the probe, issue a stop signal to the process associated with that probe instance.

Running multiple probes

You can run multiple instances of the probe.

For each running instance, specify a different port to which the server listens using the **ORBLocalPort** property.

Running multiple probes in a failover configuration

If you have implemented a peer-to-peer failover configuration, both the master probe and the slave probe have their own Object Request Broker (ORB) local port.

Where you are running this configuration in conjunction with a firewall, add the receiving port of each probe to the firewall rules. The direction of the connection is from the target system to the master or slave probe.

You set the peer-to-peer failover mode in the properties files of the master and slave probes.

Include the following peer-to-peer settings in the master's probe properties file:

```
PeerHost    :    "slave_hostname"
PeerPort    :    5555 # [communication port between master and slave probes]
```

Include the following peer-to-peer settings in the slave's probe properties file:

```
PeerHost    :    "master_hostname"
PeerPort    :    5555 # [communication port between master and slave probes]
```

Appendix A. Notices and Trademarks

This appendix contains the following sections:

- Notices
- Trademarks

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing 2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who want to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator, Department 49XA

3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, ibm.com, AIX, Tivoli, zSeries, and Netcool are trademarks of International Business Machines Corporation in the United States, other countries, or both.

Adobe, Acrobat, Portable Document Format (PDF), PostScript, and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Intel, Intel Inside (logos), MMX, and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java™ and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

IBM®