# ConsoleMonitor Tool User’s Guide

ConsoleMonitor is a command tool that specially designed for ISCS applications, has the following functions:

1. Can be used to set up ISCS run-time parameters of ISCS applications online that expose IManagedProcessCorbaDef Corba interface. So far, all ISCS agents could support this feature. All ISCS runtime parameters can be adjusted online, such as DebugLevel, DebugMaxSize, etc.
2. Can be used to set up almost all Corba runtime parameters online, such as traceLevel, traceInvocations, etc
3. Can be used to monitor and collect internal states of ISCS applications that implement ISCSInteractive Corba interface. So far, only NotificationServiceAgent could support this feature. It can be used with AutoLogCollector tool together to collect external and internal states and ISCS logs automatically, which could be much helpful to diagnose issues.
4. Can be used to set up almost all omniNotify configuration parameters online, such as DeadConProxyInterval, MaximumBatchSize, etc.

**To be noted: all command parameters’ name is case-insensitive.**

## Launch ConsoleMonitor

*ConsoleMonitor.exe --entityName=ConsoleMonitor --**RunParamFile=myapp\_start\_occ.ini*

* 1. The parameter ***RunParamFile*** is used to set parameter in initialization file, which includes some necessary and optional runtime parameter settings. Those parameters are in accordance with normal ISCS applications. In addition, you could refer to the template initialization file.
  2. Important runtime parameters:

|  |  |  |
| --- | --- | --- |
| Parameter Name | Mandatory | Comments |
| LocationKey | Yes | 1. Set the location key parameter 2. i.e. LocationKey=100 |
| NotifyHosts | No | 1. set the NotificationServiceAgent end point 2. NotifyHosts=occ-cms-01 |
| DebugLevel | No | 1. Set the debug level parameter 2. i.e. DebugLevel=INFO |
| DebugFile | No | 1. Set the debug file parameter 2. i.e. DebugFile=ConsoleMonitor.log |

* 1. Example for initialization file

--location-key=100

--notify-hosts=localhost:10000

--debug-file-max-size=100000000

--operation-mode=Control

--debug-level=CORBA

--ResolveTimeoutMs=5000

--EnableSubChangeUpdates=1

-- Collect-Corba-Log

* 1. After you launch the ConsoleMonitor, it will display the following information:

*C:\NSATest\_C955\lib\_2008>ConsoleMonitor.exe --entityName=ConsoleMonitor --RunParamFile=myapp\_start\_occ.ini --DebugFile=./log/ConsoleMonitor.log*

help

state -- show current module state

module [modulename] [entityname] [endpoint] -- switch module

module current -- list current module

module -- list all loaded modules

dosmoketest -- test message sender

quit

Corba Exception: TRANSIENT\_ConnectFailed

nsa>

The previous lines are normal commands, for example: help command, quit command, etc.

The last line is command prompt; the ***nsa*** symbol means it has connected to NotificationServiceAgent successfully after launching.

Note: When ConsoleMonitor launches, it will try to connect ***NotificationServiceAgent*** based on ***NotifyHosts*** parameter setting. If not successfully, it will report error: ***Corba Exception: TRANSIENT\_ConnectFailed***. Indeed, you can ignore the error during initialization.

## How to Connected to specific applications

Followed the command prompt ***>***, you could key in the following command

>module [modulename] [entityname] [endpoint]

* 1. Parameter description
     1. **module name**: the module type, one is ***nsa***, which used to connect NotificationServiceAgent; the other is ***common***, which used to connect other applications
     2. ***entity name***: the ISCS entity name of specific applications
     3. **end point**: Corba endpoint of the applications, its format is ipaddress:port or hostname:port, i.e. 192.168.2.100:6667, occ-cms-01:6667.
  2. How to get Corba port number

You could get port number through command ***pfiles***

* + 1. Get the correct process ID of specific applications as follows

ps –ef | grep NotifcationServcieAgent

-bash-3.00$ ps -ef | grep Notification

transact 17816 4340 0 15:12:29 pts/7 0:00 grep Notification

transact **9838** 8280 0 14:06:54 pts/5 8:30 NotificationServiceAgent --WithOmniLog --OmniToDbgLog --OmniLogLevel=40 –EndPo

**The process id is 9838**

* + 1. Get the port number from pfiles

-bash-3.00$ pfiles 9838 | grep -i AF\_INET6

sockname: AF\_INET6 :: port: **6667**

**The port number is 6667**

* 1. Example
     1. Connect / switch to OccNotificationServiceAgent

nsa>module nsa OccNotificationServiceAgent 192.168.2.100:6667

module name: nsa, entity name: OccNotificationServiceAgent, endpoint: 192.168.2.100:6667

switch module successfully!

nsa>

* + 1. Connect / switch to OccAlarmAgent

nsa>module common OccAlarmAgent 192.168.2.100:8021

module name: common, entity name: OccAlarmAgent, endpoint: 192.168.2.100:8021

switch module successfully!

common>

## Common commands

List common commands that ConsoleMonitor could process

|  |  |
| --- | --- |
| Command name | Comments |
| help | Display help, it is different based on specific module type and context environment. |
| quit | Quit ConsoleMonitor |
| module | List all supported module types. So far , only ***nsa*** and ***common*** type |
| state | Show current module state |

Example:

1. Display help

common>help

help

state -- show current module state

module [modulename] [entityname] [endpoint] -- switch module

module current -- list current module

module -- list all loaded modules

dosmoketest -- test message sender

quit

## Set up ISCS runtime parameters

1. ConsoleMonitor could set up all ISCS runtime parameters

Command format: setparam parametername=value

1. Frequent used parameters

|  |  |
| --- | --- |
| Parameter name | Comments |
| DebugLevel | Set up debug level |
| DebugFileMaxSize | Set up debug file max size |
| DebugFile | Set up debug files |

1. Example:

common>setparam DebugLevel=MSGPUB

set parameter successful!

common>

## Set up Corba runtime parameters

1. ConsoleMonitor could set up all ISCS runtime parameters

Command format: setcorbaparam parametername=value

1. Frequent used parameters

|  |  |
| --- | --- |
| Parameter name | Comments |
| traceLevel | Default value is 2, The value is cumulative, so at level 40, all trace messages are output |
| traceExceptions | Default value is 0, If the parameter is set to 1, all system exceptions are logged as they are thrown |
| traceInvocations | Default value is 0, If the parameter is set to 1, all local and remote invocations are logged. |

1. Example:
   * 1. Output Corba log

common>setcorbaparam tracelevel=40

set parameter successful!

common>setparam DebugLevel=MSGPUB

set parameter successful!

**Note: if we want to output Corba log, ISCS debug level should be set to CORBA or below.**

## Set up omniNotify configuration parameters

Before you set up OmniNotify configuration parameters, you should connect to NotificationServiceAgent, command as follows:

* 1. Set or clear flag for parameter ***DebugCosConsumerProxies***

nsa>addflag DebugCosConsumerProxies

nsa>removeflag DebugCosConsumerProxies

* 1. Set or clear flag for parameter ***DebugCosSupplierProxies***

nsa>addflag DebugCosSupplierProxies

nsa>removeflag DebugCosSupplierProxies

About the configuration parameters, you could refer to omniNotify configuration file.

## Monitor internal states of applications in interactive mode

So far, only support NotificationServiceAgent. Before you want to monitor and collect internal state of NotificationServiceAgent, you should connect to NotificationServiceAgent, command as follows:

* 1. Connect to Alarm Consumer EventChannel and output debug information

nsa>setchannel Alarm

nsa>channel go consumer

nsa>channel debug

## Monitor internal states of applications in batch mode

We could collect internal states of applications in batch mode. During batch mode, it will execute commands repeatedly

If want to enter batch mode, you should set up some parameters in initialization file.

Parameter ***BatchMode***: instruct ConsoleMonitor to enter into batch mode.

Parameter ***BatchModeLogFile***: set up the batch command file, default value is ConsoleMonitor.log.

Parameter ***BatchModeInterval***: set up the interval repeatedly, default value is 60 seconds.

Parameter ***BatchModeLogFileMaxSize***: set up the log file size, default value is 20MB

Parameter ***BatchModeCommandNumber***: set up the maximum command numbers, default value is 100.

Parameter ***BatchModeCommandn***: set up the batch command content, from 1 to n, which is smaller than ***BatchModeCommandNumber*** parameter.

For example, the initialization file can be written as follows:

--BatchMode

--BatchModeLogFile=occ-cms-01\_show\_events.log

--BatchModeLogFileMaxSize=50000000

--BatchModeCommand1="module nsa OccNotificationServiceAgent 192.168.2.100:6667"

--BatchModeCommand2="setchannel Alarm"

--BatchModeCommand3="channel go consumer"

--BatchModeCommand4="channel debug\_show\_events"

--BatchModeInterval=180