
Reference

Platform Symphony
Version 5.1
April 2011



Copyright

© 1994-2011 Platform Computing Corporation

All rights reserved.

Although the information in this document has been carefully reviewed, Platform Computing Corporation ("Platform") does not warrant it to be free of errors or omissions. Platform reserves the right to make corrections, updates, revisions or changes to the information in this document.

UNLESS OTHERWISE EXPRESSLY STATED BY PLATFORM, THE PROGRAM DESCRIBED IN THIS DOCUMENT IS PROVIDED "AS IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL PLATFORM COMPUTING BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING WITHOUT LIMITATION ANY LOST PROFITS, DATA, OR SAVINGS, ARISING OUT OF THE USE OF OR INABILITY TO USE THIS PROGRAM.

We'd like to hear from you

You can help us make this document better by telling us what you think of the content, organization, and usefulness of the information. If you find an error, or just want to make a suggestion for improving this document, please address your comments to doc@platform.com.

Your comments should pertain only to Platform documentation. For product support, contact support@platform.com.

Document redistribution and translation

This document is protected by copyright and you may not redistribute or translate it into another language, in part or in whole.

Internal redistribution

You may only redistribute this document internally within your organization (for example, on an intranet) provided that you continue to check the Platform Web site for updates and update your version of the documentation. You may not make it available to your organization over the Internet.

Trademarks

®LSF is a registered trademark of Platform Computing Corporation in the United States and in other jurisdictions.

™ACCELERATING INTELLIGENCE, PLATFORM COMPUTING, PLATFORM SYMPHONY, PLATFORM JOB SCHEDULER, PLATFORM ISF, PLATFORM ENTERPRISE GRID ORCHESTRATOR, PLATFORM EGO, and the PLATFORM and PLATFORM LSF logos are trademarks of Platform Computing Corporation in the United States and in other jurisdictions.

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

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

Microsoft is either a registered trademark or a trademark of Microsoft Corporation in the United States and/or other countries.

®Windows is a registered trademark of Microsoft Corporation in the United States and other countries.

Intel®, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Other products or services mentioned in this document are identified by the trademarks or service marks of their respective owners.

Third-party license agreements

<http://www.platform.com/Company/third.part.license.htm>

Third-party copyright notices

<http://www.platform.com/Company/Third.Party.Copyright.htm>

Contents

Part I: Commands ... 5

Symphony command summary	7
egoconfig	10
egoremoverc	15
egosetrc	16
egosetsudoers	17
egosh	18
egoshutdown	49
egostartup	50
pversions	51
rfa	52
rsdeploy	58
soamcontrol	64
soamdeploy	73
soamlog	78
soamlogoff	84
soamlogon	85
soammod	86
soamreg	90
soamshutdown	94
soamstartup	95
soamswitch	97
soamunreg	101
soamview	103
symexec	119
symping	123

Part II: Application Profile ... 131

Application profile reference	133
Consumer section	137
SOAM section	155
SessionTypes section	187
Service section	199

Part III: Daemons ... 225

egosc	227
lim	229
pem	233
pim	235
vemkd	237

Part IV: Symphony Resources ... 239

Host properties	241
Load indices	242
-R res_req	244

Part V: Configuration Files ... 249

deployment.xml	251
ego.conf	256
vem_resource.conf	264
ego.sudoers	267
ego.cluster_name.license.acct	270
ego.shared	272
ego.cluster	277

Part VI: Environment Variables ... 287

Symphony client environment variables	289
Session Director environment variables	296
EGO environment variables	297

P A R T



Commands

Symphony command summary

Platform Symphony provides commands for various purposes.

Development environment commands

The following commands are available in Symphony DE.

Command	Description
soamcontrol	Controls applications, sessions, and tasks.
soamdeploy	Deploys, removes, and displays information about service packages for consumers.
soamlog	Dynamically changes the log level for Symphony components.
soammod	Modifies session priority to enable high priority workload to finish faster.
soamreg	Registers an application or updates the application profile of a registered application.
soamunreg	Unregisters an application and deletes the application profile preventing more sessions from being started for this application.
soamstartup	In Symphony DE only, starts Symphony processes on the local host.
soamshutdown	In Symphony DE only, immediately shuts down Symphony processes on the local host.
soamswitch	Windows only. Resets the system environment for the application client: connecting to a DE cluster from the DE installation environment; connecting to a Symphony cluster from the DE installation environment; or connecting to a Symphony cluster using the Symphony installation environment.
soamview	Displays information about applications, sessions, and tasks, and displays the application profile.
symexec	Run executables as Symphony applications.
symping	Sends workload to a cluster to test and verify that Symphony components are working and responsive.

Cluster management and control commands

The following commands are available in Symphony grid.

Command	Description
egosh	Launches the administrative command interface to EGO.
egostartup	(script) Starts all EGO components of a cluster.
egoshutdown	(script) Shuts down a cluster.

Command	Description
pversions	Displays the version information for Platform products installed on a Windows host. This is a Windows command only; this command is not recognized on UNIX systems.
rfa	Transfers files between hosts.
rsdeploy	Deploys and removes middleware packages. You must be a cluster administrator to run this command.
soamlog	Dynamically changes the log level for Symphony components.

Cluster configuration commands

The following commands are available in Symphony grid.

Command	Description
egoconfig	Configures hosts.
egosetrc	Configures automatic startup of EGO on a UNIX host.
egoremoverc	Prevents automatic startup of EGO on a UNIX host.
egosetsudoers	Creates an etc/ego/sudoers file to determine accounts with root privileges on the UNIX host within a cluster.

Workload management commands

The following commands are available in Symphony grid.

Command	Description
soamcontrol	Controls applications, sessions, and tasks.
soamdeploy	Deploys, removes, and displays information about service packages for consumers.
soamlogon	Logs the user on to Platform Symphony for a specific time period.
soamlogoff	Ends the login session with Platform Symphony.
soammod	Modifies session priority to enable high priority workload to finish faster.
soamreg	Registers an application or updates the application profile of a registered application.
soamunreg	Unregisters an application and deletes the application profile preventing more sessions from being started for this application.
soamswitch	Windows only. Resets the system environment for the application client: connecting to a DE cluster from the DE installation environment; connecting to a Symphony cluster from the DE installation environment; or connecting to a Symphony cluster using the Symphony installation environment.
soamview	Displays information about applications, sessions, and tasks, and displays the application profile.

Command	Description
symexec	Run executables as Symphony applications.
symping	Sends workload to a cluster to test and verify that Symphony components are working and responsive.

egoconfig

Configures hosts.

Synopsis

egoconfig *subcommand* [*options*]

egoconfig -h

egoconfig -V

Description

Use the `egoconfig` command to join a host to the cluster, set the list of master candidates and the failover priority, and add hosts to or remove hosts from the ManagementHosts resource groups, or set other configuration options.

This is an administrative command. For most subcommands, you must be logged on as cluster administrator to issue this command.

-h

Outputs command usage and exits.

-V

Prints product version to `stderr` and exits.

Subcommand synopsis

addresourceattr "[resource *resource_name*] [resourcemap *value***resource_name*] ..."

join *master_name* [-f]

masterlist *host_name* [,*host_name*, ...]

mghost *shared_top* [-f]

mghost *shared_top* *user_account* *password* [-f]

mghost soam [-f]

mghost soam [*user_account*] [*password*] [-f]

setbaseport *base_port_no*

setlicense *license_file*

unsetmghost [-f]

addresourceattr "[resource *resource_name*] [resourcemap *value***resource_name*] ..."

Adds a resource attribute tag to the parameter `EGO_LOCAL_RESOURCES` in `ego.conf` on the local host. The attribute tag is later referenced when you create a resource group and want to add hosts to it that share the same resource attribute.

Note:

This command cannot be run on management hosts.

resource

Keyword required by EGO_LOCAL_RESOURCES to signify that the name of the resource attribute tag is boolean.

resourcemap

Keyword required by EGO_LOCAL_RESOURCES to signify that the name of the resource attribute tag is numeric. The name is preceded by a numeric value to form a name-value pair.

resource_name

Specifies the name of the resource attribute tag that identifies this type of host. For example, the resource attribute tag `scvg` could be later used to create a resource group for scavenging-ready hosts.

join *master_name* [-f]

For use on UNIX only. Adds the local UNIX host to the cluster that has the specified master host.

master_name

Specifies the host name of the master host.

-f

Suppresses confirmation of the command.

Use this option when running this command from a script.

masterlist *host_name*[,*host_name*, ...]

Specifies the list of master candidates, starting with the master host, and including all of the candidates, in the order of failover priority.

host_name

Specifies the name of the master host and each of the master candidates. Ensure that you do not specify compute hosts in this list.

Caution:

Be sure to include all master candidates in the list when you issue this command, as issuing this command overwrites the existing list.

mghost *shared_top* [-f]

For use on UNIX. Specifies to ignore the local configuration directory and look in the shared location for the configuration information and common files so the management hosts use a common set of files. Issuing this command adds this host to the management hosts resource group.

After issuing this command, you need to source your environment. Running this command creates an entry for the local host in `ego.cluster.cluster_name`.

shared_top

Specifies the path to the shared file location where configuration information is accessed by the management hosts in the cluster.

-f

Suppresses confirmation of the command.

Use this option when running this command from a script.

mghost *shared_top user_account password* [-f]

Log on using the installation OS account, or with the same permissions as the account you used to install on the host. For example, if you installed without Windows system administrator permissions, log on as cluster administrator. If you installed with Windows system administrator permissions on the local host, log on using any account that has Windows system administrator permissions on the local host.

For use on Windows. Specifies to ignore the local configuration directory and look in the shared location for the configuration information and common files so the management hosts use a common set of files. Issuing this command adds this host to the management hosts resource group.

Issuing this command changes the behavior of Windows services on this host to run under the cluster administrator account rather than the local service account. Running this command creates an entry for the local host in `ego.cluster.cluster_name`.

shared_top

Specifies the path to the shared file location where configuration information is accessed by the management hosts in the cluster.

user_account

Specifies the cluster administrator OS user account. The format is `DOMAIN\user_name`.

password

Specifies the password to use to authenticate the user account (input the actual password of the cluster administrator OS account).

-f

Suppresses confirmation of the command.

Use this option when running this command from a script.

mghost soam [-f]

For use on UNIX. Use this command when adding Platform Symphony to an existing cluster. Specifies to ignore the local Symphony configuration directory and use the EGO shared directory instead. Issue this command only on the master host so that Symphony can use shared location configurations.

After issuing this command, you need to source your environment.

-f

Suppresses confirmation of the command.

Use this option when running this command from a script.

mghost soam [*user_account*] [*password*] [-f]

Log on using the installation OS account, or with the same permissions as the account you used to install on the host. For example, if you installed without Windows system administrator permissions, log on as cluster administrator. If you installed with Windows system administrator permissions on the local host, log on using any account that has Windows system administrator permissions on the local host.

For use on Windows. Use this command when adding Platform Symphony to an existing cluster. Specifies to ignore the local Symphony configuration directory and use the EGO shared directory instead. Issue this command only on the master host so that Symphony can use shared location configurations.

user_account

Specifies the cluster administrator OS user account. The format is `DOMAIN \user_name`.

password

Specifies the password to use to authenticate the user account (input the actual password of the cluster administrator OS account).

-f

Suppresses confirmation of the command.

Use this option when running this command from a script.

setbaseport *base_port_no*

Changes the port number for the EGO_LIMservice or daemon to the specified port number.

Caution:

Shut down the cluster before issuing this command.

The remaining system port numbers are also changed as a result of issuing this command.

The following port numbers are also changed as a result of issuing this command:

- EGO_LIM_PORT
- EGO_KD_PORT
- EGO_PEM_PORT
- ESC_PORT
- REPOSITORY_SERVICE_PORT
- SD_ADMIN_PORT
- SD_SDK_PORT

base_port_no

Specifies an unused port number. The default base connection port is 7869.

EGO always uses five consecutive ports starting from the base port. By default, EGO uses ports 7869-7873.

Symphony always uses seven consecutive ports starting from the base port. By default, Symphony uses ports 7869-7875.

Valid base port numbers are between 1025 and 65531, inclusive.

setlicense *license_file*

For use on UNIX only. Copies the specified license file into the EGO configuration directory and updates the configuration file `ego.conf` with the name and location of the license file.

license_file

Specifies the full path to the license file including the file name.

unsetmghost [-f]

Log on using the installation OS account, or with the same permissions as the account you used to install on the host. For example, if you installed without Windows system administrator permissions, log on as cluster administrator. If you installed with Windows system administrator permissions on the local host, log on using any account that has Windows system administrator permissions on the local host.

Demotes the local management host to a compute host.

Specifies to look in the local configuration directory for configuration information and common files. This command cannot be run on the master host.

Before running this command, ensure the host's `lim` is not running (you may need to shut down the host first). Be sure to restart the master host after running this command for the change to take effect. Running this command removes the host entry from `ego.cluster.cluster_name`.

-f

Suppresses confirmation of the command.

Use this option when running this command from a script.

egoremoverc

Prevents automatic startup of EGO on a UNIX host.

Synopsis

egoremoverc.sh

Description

This is an administrative command. You must be logged on as root to issue this command.

Prevents automatic startup of EGO on a UNIX host when a system reboot command is issued. After this script/command is issued, EGO no longer starts automatically if the host gets rebooted. In such a case, you must manually start EGO after the host has started up.

Removes the EGO links created in the system startup directory by `egosetrc`.

egosetrc

Configures automatic startup of EGO on a UNIX host.

Synopsis

`egosetrc.sh`

Description

Configures a UNIX host to allow automatic startup of EGO on the machine when a system reboot command is issued. Creates the file `ego` under the system startup directory.

This is an administrative command. You must be logged on as `root` to issue this command.

For ease of administration, you should enable automatic startup. This starts EGO automatically when the host restarts. After running `egosetrc.sh`, perform one of the following steps:

- Restart the machine so that EGO can start up automatically, or
- Run `egosh ego start` as `root` so that the cluster runs at the expected level

Note:

If you run `egosh ego start` to start the cluster, note that the EGO service under `/var/lock/subsys` is not affected. As a result, any changes to the runlevel do not automatically affect system operations (for example, dropping from level 5 to level 2 does not automatically shut down the cluster).

If you do not configure hosts to start automatically, EGO must be started manually.

egosetsudoers

Creates an `/etc/ego.sudoers` file to determine accounts with root privileges on the UNIX host within a cluster.

Synopsis

`egosetsudoers.sh`

`egosetsudoers.sh -f | -p | -h`

Description

This command creates an `/etc/ego.sudoers` file in a UNIX host. The `/etc/ego.sudoers` file specifies accounts that are granted root privileges within a cluster. This file is owned by root and has the permissions set at 600. The default file is automatically configured to grant root privileges to the `egoadmin` account.

Note:

The cluster administrator UNIX user account is described during installation as “egoadmin” (default setting). If you indicated a different account for cluster administrator during the installation process, substitute it when you see references in the documentation to “egoadmin”. Note that this is different from the cluster administrator account that you use when you log into the Platform Management Console.

This command runs `setuid` for the `egosh` command and changes the owner of `egosh` to root.

This is an administrative command. You must be logged on as root to issue this command.

You should grant root privileges to `egoadmin` so that `egoadmin` can start a local host in the cluster and shut down or restart any hosts in the cluster from the local host. For `egoadmin` or root to start the cluster, or start any hosts specified by name, you need to be able to run `rsh` across all hosts in the cluster without having to enter a password. See your operating system documentation for information about configuring `rsh`.

-f

Used for hosts that belong to just one cluster and when the `/etc/ego.sudoers` file already exists. Changes the cluster, preserves the existing user list, removes all other contents in the file, and saves a backup copy of the old file.

-p

Used for hosts that belong to multiple clusters and the `/etc/ego.sudoers` file already exists. Adds a new cluster to the path, preserves the existing user list, removes all other contents in the file, and saves a backup copy of the old file.

egosh

Launches the administrative command interface to EGO.

Synopsis

egosh

egosh *subcommand* [*options*]

egosh -h | **help**

egosh -V

Description

With no subcommands specified, launches an interactive command console to EGO. Once it is open, you can continue to issue subcommands until you close the console.

Use the **egosh** command with one or more subcommands from within a script to run commands in batch mode.

If you want to issue administrative subcommands to control EGO objects, you must first log on to EGO using the user **l** **ogon** subcommand. You are not required to log on to view EGO objects.

-h

When running the **egosh** command in shell console mode, prints command usage to **stderr** and exits.

help

When running the **egosh** command in **egosh** console mode, displays usage information.

-V

Prints product version to **stderr** and exits.

Subcommand synopsis

Activity synopsis

activity kill [-s *signal*] -t *activity_ID* | -a *alloc_ID* | -p *consumer_name* | -c *client_name*

activity list [-l] [-f] [-t *activity_ID* | -a *alloc_ID* | -p *consumer_name* | -c *client_name* | -r *resource_name*]

activity start -a *alloc_ID* -r *resource_name* [-t *activity_name*] [-d *CWD*] [-C *CPU_limit*] [-F *file_limit*] [-D *data_limit*] [-S *stack_limit*] [-O *core_limit*] [-R *rss_limit*] [-N *nofile_limit*] [-A *as_limit*] [-e *env_name=value*;*env_name=value*...] -r *_name* [-t] [-d] [-C] [-F] [-D] [] [] [] [] [*as_limit*] [-e *env_name=value*;*env_name=*]... [] [] [OUT=*file*]] *command*
activity start -a *alloc_ID* -r *resource_name* [-t *activity_name*] [-d *CWD*] [-C *CPU_limit*] [-F *file_limit*] [-D *data_limit*] [-S *stack_limit*] [-O *core_limit*] [-R *rss_limit*] [-N *nofile_limit*] [-A *as_limit*] [-e *env_name=value*;*env_name=value*...]

activity start -a *alloc_ID* -r *resource_name* [-t *activity_name*] [-d *CWD*] [-C *CPU_limit*] [-F *file_limit*] [-D *data_limit*] [-S *stack_limit*] [-O *core_limit*] [-R *rss_limit*] [-N *nofile_limit*] [-A *as_limit*] [-e *env_name=value*;*env_name=value*...] [-f OUT=*file*;*ERR=file*] | -f *ERR=file*;*OUT=file*]]
command
activity start -a *alloc_ID* -r *resource_name* [-t *activity_name*] [-d *CWD*] [-C *CPU_limit*] [-F *file_limit*] [-D *data_limit*] [-S *stack_limit*] [-O *core_limit*] [-R *rss_limit*] [-N *nofile_limit*] [-A *as_limit*] [-e *env_name=value*;*env_name=value*...] [-f OUT=*file*;*ERR=file*] | -f *ERR=file*;*OUT=file*]] *command*

activity view [*activity_ID* ...]

Allocation synopsis

alloc diagnose [-l] -a *alloc_ID*

alloc free -a *alloc_ID* [-p *consumer_name*] [-c *client_name*]

alloc list [-l] [-a *alloc_ID*] [-p *consumer_name*] [-c *client_name*] [-r *resource_name*] [-t *activity_ID*]

alloc modify -a *alloc_ID* [-m *min_slots*] -M *max_slots* [-delta]

alloc new -p *consumer_name* -M *max_slots* [-m *min_slots*] [-a *alloc_name*] [-exclusive] [-g *resource_group*] [-s *slots_per_host*] [-R *res_req*]

alloc release -a *alloc_ID* [-block] [-autoadjust] [-modify] *host_name:nslots* ...

alloc unblock -a *alloc_ID* -n *nhosts* *host_name* ...

alloc view [*alloc_ID* ...]

Client synopsis

client list [-l] [-c *client_name*] [-a *alloc_ID*] [-p *consumer_name*] [-r *resource_name*] [-t *activity_ID*]

client reg -c *client_name* [-d *description*] [-t TTL]

client rm *client_name* ...

client unreg

client view [*client_name* ...]

client whoami

Consumer synopsis

consumer alloc [-g] [-l] [*consumer_name* ...]

consumer applyresplan [-c] [-e *error_log_directory*] *file_path*

consumer list [-l]

consumer view [*consumer_name* ...]

Debug synopsis

debug egoscon

debug egoscoff

debug limoff *host_name* ...| all

debug limon [-c *log_class*] [-p *timing_level*] [-f *log_file*] *host_name* ...| all

debug pemoff *host_name* ...| all

debug pemon [-t] [-c *log_class*] [-p *timing_level*] [-f *log_file*] [[-o "key=value"] ...] *host_name* ...| all

debug vemkdoff

debug vemkdon [-t] [-c *log_class*] [-p *timing_level*] [-f *log_file*] [[-o "key=value"]...]

EGO synopsis

ego elimrestart *env_suffix env_value* [-p *exempt_process_name*,...] [-c *close_process_name*,...][*-f*]
[*host_name* ... /all]

ego execpasswd -u *user_name* -x *password* [-noverify]

ego info

ego restart [*-f*] [*host_name* ... | all]

ego shutdown [*-f*] [*host_name* ... | all]

ego start [*-f*] [*host_name* ... | all]

License synopsis

license info

Quit synopsis

quit

Resource synopsis

resource close [-reclaim] [*-c comment*] *resource_name* ...

(**resource group**) | **rg** [*-l*] [*group_name* ...]

resource list [*-l*] [*-m* | *-s* | *-t* | *-a* | *-o attribute*,...] [*-R res_req*] [*resource_name* ...]

resource open *resource_name* ...

resource remove *host_name* ...

resource view [*resource_name* ...]

Service synopsis

service list [*-l*] [*-s service_name*] [*-a alloc_ID*] [*-p consumer_name*] [*-r resource_name*]

service start *service_name* ... | all

service stop *service_name* ... | all

service view [*service_name* ...]

User synopsis

user add -u *user_account* -x *password* [-e *email*] [-t *telephone*] [-d *description*]

user assignrole -u *user_account* -r *role* [-p *consumer_name*]

user delete -u *user_account*

user list [*-l*]

user logoff

user logon -u *user_account* -x *password*

user modify -u *user_account* [-x *password*] [-e *email*] [-t *telephone*] [-d *description*]

user roles4user -u *user_account*

user users4role -r role [-p consumer_name]

user unassignrole -u user_account -r role [-p consumer_name]

user view [user_account ...]

activity kill [-s signal] -t activity_ID | -a alloc_ID | -p consumer_name | -c client_name

Terminates all activities for the specified allocation, consumer or client.

-s signal

Sends a specific signal.

Signals are operating-system dependent. Refer to the list of signals available for your operating system.

-t activity_ID

Specifies the activity to terminate.

-a alloc_ID

Specifies the allocation to which this action applies.

-p consumer_name

Terminates all activities for the specified consumer.

Specify the unique consumer name, or the full path to the consumer name. Specify as many levels within the consumer tree as required to uniquely identify the consumer, as follows:

/top-level_consumer_name/. . . /leaf_consumer_name

-c client_name

Specifies the name of the client to which this action applies.

activity list [-l] [-f] [-t activity_ID | -a alloc_ID | -p consumer_name | -c client_name | -r resource_name]

Lists all activities in the cluster.

-l

Provides the same information with a longer name field, if some are truncated when -l is not specified.

-f

Lists the failed activities. These activities are in the finish state and the exit reason is not "None", "Terminated by SIGKILL", "Terminated by job controller" or "Terminated by SIGKILL, job controller does not exist or failed".

-t activity_ID

Specifies the activity to list.

-a alloc_ID

Lists all activities that belong to the specified allocation.

-p consumer_name

Lists all activities for the specified consumer.

Specify the unique consumer name, or the full path to the consumer name. Specify as many levels within the consumer tree as required to uniquely identify the consumer, as follows:

/top-level_consumer_name/.../leaf_consumer_name

-c client_name

Specifies the name of the client to which this action applies.

-r resource_name

Lists all the activities that are using the specified resource.

activity start -a *alloc_ID* -r *resource_name* [-t *activity_name*] [-d CWD] [-C CPU_limit] [-F *file_limit*] [-D *data_limit*] [-S *stack_limit*] [-O *core_limit*] [-R *rss_limit*] [-N *nofile_limit*] [-A *as_limit*] [-e *env_name=value*;*env_name=value*] ...] *command*

Starts an activity on the specified host.

-a alloc_ID

Specifies the allocation this activity belongs to.

-r resource_name

Specifies the host on which to start the activity.

-t activity_name

Specifies to start the activity using the name specified.

-d CWD

Specifies the current working directory from which the activity is started.

If you do not specify a directory, /tmp is used on UNIX systems, and %TEMP% is used on Windows systems.

-C CPU_limit

Specifies the maximum amount of CPU time this activity may use before being terminated by the system.

After specifying a value, specify the units for measuring CPU time:

- **s**: seconds. For example, 20s specifies a CPU limit of 20 seconds.

- **m**: minutes. For example, **40m** specifies a CPU limit of 40 minutes.
- **h**: hours. For example, **2h** specifies a CPU limit of two hours.

-F file_limit

Specifies the maximum file size this activity may use before being terminated by the system.

After specifying a maximum file size, specify one of the following values:

- **b**: bytes. For example, **400b** specifies a limit of 400 bytes.
- **k**: kilobytes. For example, **40k** specifies a limit of 40 kilobytes.
- **m**: megabytes. For example, **4m** specifies a limit of 4 megabytes.
- **g**: gigabytes. For example, **4g** specifies a limit of 4 gigabytes.

-D data_limit

Specifies the maximum data segment size limit for each of the processes belonging to the activity. If this limit is exceeded, the activity is terminated by the system.

After specifying a value, specify the units for measuring the data limit:

- **b**: bytes. For example, **400b** specifies a limit of 400 bytes.
- **k**: kilobytes. For example, **40k** specifies a limit of 40 kilobytes.
- **m**: megabytes. For example, **4m** specifies a limit of 4 megabytes.
- **g**: gigabytes. For example, **4g** specifies a limit of 4 gigabytes.

-S stack_limit

Specifies the maximum stack segment size for each of the processes belonging to the activity. If this limit is exceeded, the activity is terminated by the system.

After specifying a value, specify the units for measuring the stack limit:

- **b**: bytes. For example, **400b** specifies a limit of 400 bytes.
- **k**: kilobytes. For example, **40k** specifies a limit of 40 kilobytes.
- **m**: megabytes. For example, **4m** specifies a limit of 4 megabytes.
- **g**: gigabytes. For example, **4g** specifies a limit of 4 gigabytes.

-O core_limit

Specifies the maximum core file size for all the processes belonging to the activity. If this limit is exceeded, the activity is terminated by the system.

After specifying a value, specify the units for measuring the core size limit:

- **b**: bytes. For example, **400b** specifies a limit of 400 bytes.
- **k**: kilobytes. For example, **40k** specifies a limit of 40 kilobytes.
- **m**: megabytes. For example, **4m** specifies a limit of 4 megabytes.
- **g**: gigabytes. For example, **4g** specifies a limit of 4 gigabytes.

-R rss_limit

Specifies the maximum resident set size, limiting physical memory usage for each process belonging to the activity. If this limit is exceeded, the activity is terminated by the system.

After specifying a value, specify the units for measuring the physical memory limit:

- **b**: bytes. For example, 400b specifies a limit of 400 bytes.
- **k**: kilobytes. For example, 40k specifies a limit of 40 kilobytes.
- **m**: megabytes. For example, 4m specifies a limit of 4 megabytes.
- **g**: gigabytes. For example, 4g specifies a limit of 4 gigabytes.

-N nofile_limit

Specifies the maximum number of open file descriptors this activity may use.

-A as_limit

Specifies the maximum process size (address space) for each process belonging to the activity.

After specifying a value, specify the units for measuring the address space limit:

- **b**: bytes. For example, 400b specifies a limit of 400 bytes.
- **k**: kilobytes. For example, 40k specifies a limit of 40 kilobytes.
- **m**: megabytes. For example, 4m specifies a limit of 4 megabytes.
- **g**: gigabytes. For example, 4g specifies a limit of 4 gigabytes.

-e env_name=value ...

Sets the environment variables for this activity. Specify as many environment variable/value pairs as required to define the environment.

To specify multiple environment variable/value pairs, separate the pairs with a space.

-f OUT=file[;ERR=file] | -f ERR=file[;OUT=file]

Redirects standard out and standard error of the activity to the specified files.

Specify an absolute or relative path. If you specify a relative path, the path will be relative to the execution working directory of the activity process creating/opening the file.

If you use relative paths on Windows, the path will be relative to the execution working directory of the parent PEM. ????

If the files do not exist, they are created. If the files exist, information is appended to the files. Once the files are created, it is your responsibility to maintain them.

For file naming limitations and conventions, refer to the specific operating-system documentation. In addition, take into account the following when naming files:

- Do not use semi-colon (;) or quotation marks (") in the file name.
- Do not use UNC paths. For example, //hostA/mountpoint/myfile. You will need to mount the network accessible mount point as a local drive instead of using an UNC path.
- Spaces in file names are allowed as long as the whole argument is enclosed in quotes. For example: -f "IN=myfile;OUT=outfile".

command

Specifies the command to run.

The command to run must always be specified last.

activity view [*activity_ID ...*]

Displays detailed information about the activities in the cluster, including its resources, allocations, current status, start time, and so on.

activity_ID ...

Specifies the ID for the activity for which you want detailed information.

alloc free -a *alloc_ID* | -p *consumer_name* | -c *client_name*

Frees all allocations for the specified consumer or client, returning resources to the cluster and removing the allocation names.

-a alloc_ID

Specifies the ID of the allocation to free.

-p consumer_name

Frees all activities for the specified consumer.

Specify the unique consumer name, or the full path to the consumer name. Specify as many levels within the consumer tree as required to uniquely identify the consumer, as follows:

/top-level_consumer_name/.../leaf_consumer_name

-c client_name

Specifies the name of the client to which the allocation was made.

alloc list [-l] [-a *alloc_ID*] [-p *consumer_name*] [-c *client_name*] [-r *resource_name*] [-t *activity_ID*]

Lists all allocations in the cluster, listing the allocation ID, consumer, client, resource groups and resources used by each allocation.

-l

Provides the same information with a longer name field, if some are truncated when -l is not specified.

-a alloc_ID

Lists the allocated resources for the specified allocation.

-p consumer_name

Lists all activities for the specified consumer.

Specify the unique consumer name, or the full path to the consumer name. Specify as many levels within the consumer tree as required to uniquely identify the consumer, as follows:

/top-level_consumer_name/.../leaf_consumer_name

-c client_name

Lists the resources allocated to the specified client.

-r resource_name

Lists all allocations that include the specified resource.

-t activity_ID

Lists all allocations that include the specified activity.

alloc modify -a *alloc_ID* [-m *min_slots*] -M *max_slots* [-delta]

Requests an increased number of resources for an existing allocation.

-a alloc_ID

Specifies the ID of the allocation to change.

-m min_slotsSpecifies the minimum number of slots to be allocated, or the minimum additional slots to be allocated, depending on if `-delta` is specified.**-M max_slots**Specifies the maximum number of slots to be allocated, or the maximum additional slots to be allocated, depending on if `-delta` is specified.**-delta**

Specifies that the minimum and maximum slots requested are in addition to the existing allocation for this consumer.

alloc new -p *consumer_name* -M *max_slots* [-m *min_slots*] [-a *alloc_name*] [-exclusive] [-g *resource_group*] [-s *slots_per_host*] [-R *res_req*]

Requests a new resource allocation for the specified consumer from the specified resource group.

-p consumer_name

Specifies the consumer to allocate the resources to.

Specify the unique consumer name, or the full path to the consumer name. Specify as many levels within the consumer tree as required to uniquely identify the consumer, as follows:

*/top-level_consumer_name/.../leaf_consumer_name***-M max_slots**

Specifies the maximum number of slots to be allocated.

-m min_slots

Specifies the minimum number of slots to be allocated.

-a alloc_name

Specifies a name to identify the new allocation request.

Specify a name that is unique within the cluster. Specify up to 40 alphanumeric characters.

-exclusive

Specifies that this allocation request is for the exclusive use of these resources by this consumer.

Note that a host may still be distributed to several allocations if it appears in multiple host groups, despite indicating exclusive usage.

-g resource_group

Specifies the resource group from which to allocate resources.

-s slots_per_host

Specifies the number of slots per host required (on both single- and multi-CPU hosts).

-R res_req

Specifies the resource requirement to use to select the most appropriate host for this allocation.

Specify name value pairs for the resource requirement(s). Multiple resource requirements are separated with the characters **&&**.

Important:

If the command is issued in whole from the shell console or the requirement has white space in it, enclose the requirement in double quotation marks. For example:

```
>egosh resource list -R "select(mem>100 && it>1) "
```

If the command is issued from the egosh console, do not use quotation marks. For example:

```
>egosh
```

```
>resource list -R select(mem>100)
```

alloc release -a *alloc_ID* [-block] [-autoadjust] [-modify] *host_name:nslots* ...

Reduces an allocation by the specified number of hosts or slots.

-a alloc_ID

Specifies the ID of the allocation from which to release the slots.

-block

Releases the slots and prevents this host from being allocated to this consumer again.

Use this option if a host is not behaving properly. You can reverse this option later using the `alloc unblock` subcommand.

-autoadjust

Automatically adjusts the allocation request to match the current number of slots. This prevents the resources from being assigned back to the current allocation.

Issuing this command without specifying a number of slots removes any unfulfilled slot requests for this allocation, and modifies the request to the current number of slots.

Use this option when you do not expect to need the slots anymore.

-modify

Automatically decrements the allocation request by the number of slots being released. The `-autoadjust` option takes precedence over the `-modify` option.

host_name:nslots ...

Releases the specified number of slots from the specified hosts.

Specify the name of the host followed by the number of slots to release from that host.

To specify multiple hosts and numbers of slots, separate the host and slot combinations with a space.

`alloc unblock -a alloc_ID -n nhosts host_name ...`

Specifies to allow blocked hosts to be allocated to this consumer again. Use this command to undo a previous `alloc release -block` subcommand.

-a *alloc_ID*

Specifies the ID of the allocation from which to unblock the host.

-n *nhosts*

Specifies the number of hosts to unblock, allowing the hosts to be allocated to this consumer again.

***host_name* ...**

Specifies the host names to unblock.

To specify multiple hosts, separate the hosts with a space.

`alloc view [alloc_ID ...]`

Displays detailed information about all allocations, including the allocation ID, current users, consumer, resource groups, resource requirements, minimum and maximum slots requested, whether it has exclusive use of the host, names of the allocated hosts, and any blocked hosts.

***alloc_ID* ...**

Displays information about the specified allocation.

client list [-l] [-c *client_name*] [-a *alloc_ID*] [-p *consumer_name*] [-r *resource_name*] [-t *activity_ID*]

Displays a list of the registered clients in the cluster, and information about each client, including the host name and port number, the channel, and whether the client is connected. Client names are truncated to 12 characters.

-l

Provides the same information with a longer name field, if some are truncated when **-l** is not specified.

-c *client_name*

Specifies the client to which this action applies.

-a *alloc_ID*

Lists the client who has allocated to the specified allocation.

-p *consumer_name*

Lists all the clients for the specified consumer.

-r *resource_name*

Lists all the clients that are using the specified resource.

-t *activity_ID*

Lists the client that has started the specified activities.

client reg -c *client_name* [-d *description*] [-t TTL]

Registers the current EGO client with the system so that it can start sending requests to EGO for resources. Following registration, the client may be assigned allocations. The client is assigned a unique identifier such as `autoAssignedClient 0` or `autoAssignedClient 1`.

-c *client_name*

Specifies to register the client with a specific identifier.

Specify a name that is unique within the cluster. Specify up to 40 ASCII characters.

-d *description*

Specifies a description for the client being registered. This description appears with the information displayed using the `client view` subcommand. Enclose description in quotation marks if there are spaces within it.

-t TTL

Specifies the client TTL (time to live) in minutes. If the option is not set, default TTL is 900 minutes.

client rm *client_name* ...

Removes and unregisters the specified client from the system. Use this command to remove a client that is not responding.

client_name ...

Specifies the name of the client to be removed.

client unreg

Unregisters the current client from the system. Once this operation completes, the client can no longer request resources from EGO.

After unregistration, all allocations to this client are released.

client view [*client_name ...*]

Displays a list of the registered clients in the cluster, and information about each client, including the host name and port number, the channel, and whether the client is connected.

client_name ...

Specifies the name of one or more clients you want to view.

client whoami

Prints the user name associated with the current client.

consumer alloc [-g] [-l] [*consumer_name ...*]

Displays allocation and demand information for each leaf consumer.

-g

Provides resource group details.

-l

Provides the same information with a longer name field, if some are truncated when -l is not specified.

consumer_name

Specifies the name of the consumer(s) for which you want to display information.

consumer applyresplan [-c] [-e *error_log_directory*] *file_path*

Applies the resource plan specified in the file path. Once you apply it, the plan is in effect immediately.

This is an administrative subcommand. You must first log on as cluster administrator or consumer administrator before you can issue this subcommand.

-c

Checks the resource plan for validity and well formedness without applying it.

-e error_log_directory

Specifies the directory where stderr with error messages outputs.

file_path

Specifies the resource plan file you want in effect immediately. The file must be XML, valid, and well-formed. If it is rejected for any reason, the previously applied resource plan stays in effect.

consumer list [-l]

Displays a list of the full paths to the consumers in the cluster, and lists the administrators assigned to each consumer.

-l

Provides the same information with a longer name field, if some are truncated when **-l** is not specified.

consumer view [consumer_name ...]

Displays a list of the consumers in the cluster, and detailed information about each consumer, including the administrators assigned to that consumer and the resource policies applied to each consumer.

consumer_name

Displays information about the specified consumer.

Specify the unique consumer name without the full path or, if it is not unique, specify the full path to the consumer name.

/top-level_consumer_name/. . . /leaf_consumer_name

debug limoff host_name ... | all

Turns off debugging of the `lim` daemon on the specified hosts.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

host_name

Turns off debugging of the `lim` daemon on the specified host.

all

Turns off debugging of the `lim` daemon on all hosts in the cluster.

debug limon [-c log_class] [-p timing_level] [-f log_file] host_name ... | all

Turns on debugging of the `lim` daemon to `LOG_DEBUG` level on the specified host.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

-c log_class

Specifies a log class, which limits the messages collected to specific types, or limits debugging to specific components. Use this option to filter out and reduce the amount of data kept. For a complete list of log classes, refer to the *Troubleshooting* section of the *Cluster and Application Management Guide*.

To specify multiple log classes, separate the log classes with a space, and enclose the string in double quotes.

-p timing_level

Specifies function performance timing level to specify the number of layers of components to measure the time a process takes. Specify a number from 1 (time the process at the top component level) to 5 (time the processes at five layers depth). If no value is specified, timing is disabled.

-f log_file

Specifies the path and file name to where the log files are to be written. For example, if you specify **-f /tmp/debuglog**, the messages are logged to `/tmp/debuglog.lim.log.hostname`

If you do not specify a file name and path, defaults to the current log file.

host_name

Turns on debugging of the `lim` daemon on the specified host.

all

Turns on debugging of the `lim` daemons on all hosts in the cluster.

debug pemoff *host_name ... | all*

Turns off debugging of the `pem` daemon on the specified hosts.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

host_name

Turns off debugging of the `pem` daemon on the specified host.

all

Turns off debugging of the `pem` daemon on all hosts in the cluster.

debug pemon [-t] [-c *log_class*] [-p *timing_level*] [-f *log_file*] [[-o "*key=value*" ...] *host_name ... | all*

Turns on debugging of the `pem` daemon to LOG_DEBUG level on the specified host.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

-t

Sets the logging level to LOG_TRACE, which logs all program steps.

-c log_class

Specifies a log class, which limits the messages collected to specific types, or limits debugging to specific components. Use this option to filter out and reduce the amount of data kept. For a complete list of log classes, refer to the *Troubleshooting* section of the *Cluster and Application Management Guide*.

To specify multiple log classes, separate the log classes with a space, and enclose the string in double quotes.

-p timing_level

Specifies function performance timing level to specify the number of layers of components to measure the time a process takes. Specify a number from 1 (time the process at the top component level) to 5 (time the processes at five layers depth). If no value is specified, timing is disabled.

-f log_file

Specifies the path and file name to where the log files are to be written. For example, if you specify **-f /tmp/debuglog**, the messages are logged to `/tmp/debuglog.pem.log.hostname`

If you do not specify a file name and path, defaults to the current log file.

-o "key=value"

Specifies the debug object class and identifier. The format is *key=value*, where valid values for *key* are ACTIVITY and ALLOC and *value* is the ID of the activity or allocation.

To specify multiple key and value pairs, specify **-o** for each object class and separate the options with a space.

host_name

Turns on debugging of the `pem` daemon on the specified host.

all

Turns on debugging of the `pem` daemons on all hosts in the cluster.

debug vemkdoff

Turns off dynamic debugging of the EGO kernel daemon `vemkd`.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

debug vemkdon [-t] [-c log_class] [-p timing_level] [-f log_file] [[-o "key=value"] ...]

Turns on dynamic debugging of the EGO kernel daemon `vemkd` to LOG_DEBUG level.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

-t

Sets the logging level to LOG_TRACE, which logs all program steps.

-c log_class

Specifies a log class, which limits the messages collected to specific types, or limits debugging to specific components. Use this option to filter out and reduce the amount

of data kept. For a complete list of log classes, refer to the *Troubleshooting* section of the *Cluster and Application Management Guide*.

To specify multiple log classes, separate the log classes with a space and enclose the string in double quotes.

-p timing_level

Specifies function performance timing level to specify the number of layers of components to measure the time a process takes. Specify a number from 1 (time the process at the top component level) to 5 (time the processes at five layers depth). If no value is specified, timing is disabled.

-f log_file

Specifies the path to where the log files are to be written. For example, if you specify **-f /tmp/debuglog**, the messages are logged to `/tmp/debuglog.vemkd.log.hostname`

If you do not specify a file name and path, defaults to the current log file.

-o "key=value"

Specifies the debug object class and identifier. The format is *key=value*, where valid values for *key* are ACTIVITY and ALLOC and *value* is the ID of the activity or allocation.

To specify multiple key and value pairs, specify **-o** for each object class and separate the options with a space.

ego execpasswd -u *user_name* -x *password* [-noverify]

Registers and verifies the password for a Windows execution user account.

Registering the password allows EGO to use the account to run work on Windows hosts.

This is an administrative command. You must be cluster administrator to issue this command. In addition, to verify the password, you must be logged on to Windows as the OS account administrator, `egoadmin`.

-u user_name

Specifies the fully-qualified Windows user name of the execution account to register the password for.

-x password

Specifies the password to register for the Windows execution user account.

-noverify

Registers the password without verification. This option is required if you run this command from a UNIX host. Only a Windows host can verify this password.


```
ego elimrestart env_suffix env_value [-p
exempt_process_name;...] [-c close_process_name;
...] [-f] [host_name ... | all]
```

Restarts/reconfigures external load information manager(s) with an environment variable (*elim.sa*). Generally used for host scavenging feature. During restart, the *lim* passes along configuration information to the scavenging agent such as thresholds of resources that are used to evaluate trigger conditions, whether the host is currently enabled for scavenging, and whether the grace period is disabled.

Note:

After running this command, it takes several seconds for the new configuration to take effect, dependent upon how frequently EGO refreshes host information (as set in *EGO_RESOURCE_UPDATE_INTERVAL* in *ego.conf*).

You must be logged on to Windows as the local systems OS account administrator, or logged on to Linux as the root OS account.

env_suffix

Always specify **SA** (scavenging agent) to indicate the host scavenging feature.

env_value

Specifies whether to enable host scavenging with grace period (on), enable host scavenging without grace period (fastrelease), or disable (off) host scavenging on this host.

Specifies the thresholds of load indices used to evaluate host workload and to trigger host scavenging.

Enter the environment value in this format, delimited by commas without any spaces:
<scavenging_flag>,<user_idle_time_threshold_in_minutes>,<Adjusted_CPU_utilization_threshold_in_percentage>,<CPU_idle_time_threshold_in_minutes>.

For example:

```
egosh ego elimrestart SA fastrelease, 2, 0.3, 1.67 all
```

This example enables (turns “on”) host scavenging on all hosts, disables the grace period, sets the user idle time threshold (*uit_t*) to 2 minutes, Adjusted CPU utilization threshold (*cu_t*) to 30%, and CPU idle time threshold (*ci_t_t*) to 1.67 minutes (or 100 seconds).

Note:

Threshold values are specified by numbers greater than zero. They do not need to be whole numbers.

-c close_process_name

Specifies processes that will cause the host to be closed if the processes are running. A process name can be specified either with or without a path; if specified without a path,

it must be unique; if specified with a path, the path cannot include spaces. The list of process names is limited to 256 characters.

-p exempt_process_name

Specifies the processes to be excluded from the calculation of Adjusted CPU utilization for one or more hosts. A process name can be specified either with or without a path; if specified without a path, it must be unique; if specified with a path, the path cannot include spaces. The list of process names is limited to 256 characters.

-f

Executes command immediately without asking for confirmation. Use this option when you are issuing `egosh ego eli mrestart` from within a script and do not want the script to stop running to respond to prompts.

host_name ...

Specifies the name of the host or hosts on which to restart/reconfigure the external load information manager(s).

To specify multiple hosts, separate the host names with a space.

If no host name is given, then control is assumed to be local.

all

Restarts/reconfigures the external load information manager (`eli m`) on all hosts in the cluster.

ego info

Displays information about the cluster, including the cluster name, the name of the master host, and the version of EGO.

ego restart [-f] [*host_name ...* | *all*]

Restarts EGO on the local host, or, if issued from the master host, restarts EGO on all the hosts in the cluster. Does not affect running work or services.

This is an administrative subcommand. On UNIX, you must be logged on with `root` permissions to issue this command. On Windows, you must be logged on as cluster administrator to issue this command.

-f

Forces the action on the host without validating the configuration file. Use this option when you are issuing the command from within a script and do not want the script to stop running to respond to prompts.

host_name ...

Specifies the name of the host or hosts on which to restart EGO.

To specify multiple hosts, separate the host names with a space.

You cannot use this option from a compute host unless the master host is up and running.

all

Restarts EGO on all hosts in the cluster.

You cannot use this option from a compute host unless the master host is up and running.

ego shutdown [-f] [*host_name* ... | all]

Stops EGO on the local host, or, if issued from the master host, stops EGO on all the hosts in the cluster.

This is an administrative subcommand. On UNIX, you must be logged on with `root` permissions to issue this command. On Windows, you must be logged on as cluster administrator to issue this command.

-f

Forces the action on the host without validating the configuration file. Use this option when you are issuing the command from within a script and do not want the script to stop running to respond to prompts.

host_name ...

Specifies the name of the host or hosts on which to stop EGO.

To specify multiple hosts, separate the host names with a space.

You cannot use this option from a compute host unless the master host is up and running.

all

Stops EGO on all hosts in the cluster.

Caution:

Never use this option to shut down the cluster. To shut down the entire cluster, run the `egoshutdown` command.

You cannot use this option from a compute host unless the master host is up and running.

ego start [-f] [*host_name* ... | all]

Starts EGO on the local host or, if issued from the master host, starts EGO on all the hosts in the cluster.

This is an administrative subcommand. On UNIX, you must be logged on with `root` permissions to issue this command. On Windows, you must be logged on as cluster administrator to issue this command.

-f

Forces the action on the host without validating the configuration file. Use this option when you are issuing the command from within a script and do not want the script to stop running to respond to prompts.

host_name ...

Specifies the name of the host or hosts on which to start EGO.

To specify multiple hosts, separate the host names with a space.

You cannot use this option from a compute host unless the master host is up and running.

To use this option on UNIX, you must have `root` permission on each host and have `rsh` configured for your account for each host. You may need to add an entry for the local host in the `.rhosts` file for `root`.

Note:

You cannot start EGO on a UNIX host from a Windows host, or vice versa.

all

Starts EGO on all hosts in the cluster. Use this option when you want to start the entire cluster.

You cannot use this option from a compute host unless the master host is up and running.

To use this option on UNIX, you must have `root` permission on each host and have `rsh` configured for your account. You may need to add an entry for the local host in the `.rhosts` file for `root`.

Note:

You cannot start EGO on UNIX hosts from a Windows host, or vice versa.

license info

Displays the number of supported cores, the type of base license installed in the cluster, and the license status of add-on products.

quit | q

Closes the interactive command console. If you are logged on to EGO, `quit` does not log you off when it closes the command console. Alias: `q`.

resource close [-reclaim] [-c *comment*] *resource_name* ...

Closes a resource, preventing further allocation. Closing a resource does not change its allocation status. If the resource is currently allocated to a consumer, the resource remains allocated until the consumer returns it voluntarily. If the resource is not currently allocated to a consumer, the resource remains in its unallocated state. Existing workload finishes running before closing.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

-reclaim

EGO reclaims the host before it closes; running workload terminates as per the configured grace period. The host is prevented from further allocation. If the resource

is currently allocated to a consumer, it is reclaimed. Once reclaimed, it is not allocated to another consumer.

After issuing this command, the host status changes to CLOSED; the reported reason is “cluster administrator closes and reclaims host”.

-c comment

Specifies a reason why this action was requested. The reason can comprise up to 1024 alphanumeric or special characters, except control characters (Ctrl + key) and multi-byte characters. The description must be enclosed in double quotes (" ") if it contains spaces.

resource_name ...

Specifies the name of the resource or resources to close.

To close multiple resources, separate the resource names with a space.

(resource group) | rg [-l] [*group_name ...*]

Displays information about all of the resource groups in the cluster including the number of hosts in the group, the total number of slots, the number of free and allocated slots, and detailed usage information describing distribution among consumers.

Note:

This subcommand limits the displayed length of the resource group name to 21 characters. If the full length of the resource group name is required, use `egosh rg -l` instead.

rg

Is an alias to the `resource group` subcommand. You can use this as a shortcut instead of typing the full subcommand name.

- **ALLOCATED:** Indicates the total number of resources allocated to a consumer.
- **FREE:** Indicates the total number of unused resources, including unused owned and unused shared (guaranteed), as per the resource plan
- **OWN:** Indicates the configured ownership numbers, as per the resource plan.
- **SHARE:** Indicates the configured share percentage among siblings, as per the resource plan.

-l

Lists values for allocated and free slots within resource groups. Detailed usage information includes breakdown of owned, shared, and borrowed slots (both in-use and unused slots) in the cluster:

- **OWN_USE:** Indicates number of owned resources assigned to consumer.
- **SHARE_USE:** Indicates number of resources assigned to consumer from share pool.
- **BORROW_USE:** Indicates number of resources borrowed from other consumers.
- **OWN_FREE:** Indicates number of remaining (unused) owned resources as guaranteed from resource plan.

- **SHARE_FREE:** Indicates number of remaining (unused) share pool resources as guaranteed from resource plan.

Note:

Values for OWN_FREE and SHARE_FREE may not add up to the actual "free" or total number of resources for the resource group. Some resources reflected in the number may be reclaimed resources.

group_name

Specifies the name of the resource group for which you want information displayed. For example, `ManagementHosts`.

resource list [-l] [-m | -s | -t | -a | -o *attribute*,...] [-R *res_req*] [*resource_name* ...]

Displays information about the resources in the cluster, listing each host and information about the resources on each host.

-l

Provides the same information with a longer name field, if some are truncated when `-l` is not specified.

-m

Displays the list of failover candidate hosts in the cluster and identifies which host is currently the master.

-s

Displays summaries of the hosts in the cluster, including information on host states and resource utilization.

-t

Displays a list of host types defined in the cluster.

-a

Displays all load indices for all resources.

-o *attribute*,...

Specifies the attributes to include in the display. Use this option to customize the output, including only those attributes you are interested in. For example:

resource list -o status,type,ncpus

Specify one (or more) of the following:

- `status`: Current state of the host
- `type`: Type of host
- `ncpus`: Number of CPUs as seen by EGO (value used to determine the number of slots; can be overridden by resource group configuration)

- `nprocs`: Number of physical processors (if `ncpus` defined as `procs`, then `ncpus = nprocs`)
- `ncores`: Number of cores per processor (if `ncpus` defined as `cores`, then `ncpus = nprocs * ncores`)
- `nthreads`: Number of threads per core (if `ncpus` defined as `threads`, then `ncpus = nprocs * ncores * nthreads`)
- `ut`: CPU utilization
- `mem`: Available memory
- `swp`: Available swap space
- `pg`: Paging rate
- `io`: Disk I/O rate
- `slots`: Number of slots
- `free_slots`: Number of free slots
- `r15s`: 15-second load
- `r15m`: 15-minute load
- `r1m`: 1-minute load
- `model`: The host model
- `cpuf`: The CPU factor
- `maxmem`: Maximum memory
- `maxswp`: Maximum swap space
- `tmp`: Available temp space
- `maxtmp`: Maximum space in `/tmp`
- `ndisks`: Number of local disks
- `it`: Idle time
- `ls`: Logon users
- `resourceattr`: Resource attributes assigned to this host
- `processpri`: The OS process priority of cluster workloads (either `normal` or `lowest`)
- *resource_name*: The name of a host resource

Note:

You cannot use this command option to view global ncpu settings. This information can only be viewed directly in the shared copy of `ego.conf`.

-R res_req

Displays information about the resources that match the resource requirement string specified.

Specify name value pairs for the resource requirement(s). Multiple resource requirements are separated with the characters **&&**.

Important:

If the command is issued in whole from the shell console or the requirement has white space, enclose the requirement in double quotation marks. For example:

```
>egosh resource list -R "select(mem>100 && it>1) "
```

If the command is issued from the egosh console, do not use quotation marks. For example:

```
>egosh
```

```
>resource list -R select(mem>100)
```

resource _name ...

Specifies the name of the resource you want to list.

Displays information about the resource with the specified name.

resource open *resource_name* ...

Opens the specified resource, allowing it to accept requests.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

resource _name ...

Specifies the name of the resource or resources to open.

To open multiple resources, separate the resource names with a space.

resource remove *host_name* ...

Removes the specified host from the cluster. EGO is also shut down if the host is closed. To remove a host, it must have joined the cluster dynamically and is now either unavailable or closed without running workload.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

host _name ...

Specifies the name of the host or hosts to remove.

To remove multiple hosts, separate the host names with a space.

resource view [*resource_name* ...]

Displays all the information about all resources.

resource _name ...

Specifies the name of the resource or resources you want to view.

Displays information about the specified resource or resources.

To view multiple resources, separate the resource names with a space.

service list [-l] [-s *service_name*] [-a *alloc_ID*] [-p *consumer_name*] [-r *resource_name*]

Lists registered service(s) defined in EGO service controller.

-l

Provides the same information with a longer name field, if some are truncated when **-l** is not specified.

-s *service_name*

Specifies the service to which this action applies.

-a *alloc_ID*

Lists all services that belong to the specified allocation.

-p *consumer_name*

Lists all the services for the specified consumer.

-r *resource_name*

Lists all the services that are using the specified resource.

service start *service_name* ... | all

Starts registered service(s) defined in EGO service controller. If this is a service that is configured to start automatically, enables the service to be started automatically.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

***service_name* ...**

Starts the specified service(s).

all

Starts all registered services.

service stop *service_name* ... | all

Stops registered service(s) defined in EGO service controller.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

***service_name* ...**

Stops the specified service(s).

all

Stops all registered services.

service view [*service_name* ...]

Displays registered service(s) defined in EGO service controller.

service_name ...

Displays information about the specified service(s).

user add -u *user_account* -x *password* [-e *email*] [-t *telephone*] [-d *description*]

Creates a new user account in the EGO user database with the specified name.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

-u *user_account*

Specifies the name of the user account to create.

Specify a unique name with up to 32 alphanumeric or special characters, except greater than (>), less than (<), ampersand (&), or control characters (Ctrl + key).

-x *password*

Specifies the password to be used to authenticate the user when this user account is accessed.

Specify one to eight alphanumeric or special characters, except greater than (>), less than (<), ampersand (&), or control characters (Ctrl + key).

-e *email*

Specifies the email address of the user to whom this account belongs.

Specify up to 64 alphanumeric or special characters, except greater than (>), less than (<), ampersand (&), or control characters (Ctrl + key).

-t *telephone*

Specifies the telephone number of the user to whom this account belongs.

Specify up to 20 numbers and spaces.

-d *description*

Specifies any additional information about the user account or the user to whom this account belongs.

Specify up to 200 alphanumeric or special characters, except control characters (Ctrl + key). Enclose description in quotation marks if there are spaces within it.

user assignrole -u *user_account* -r *role* [-p *consumer_name*]

Assigns the specified role to the specified user account, and optionally specifies the consumer this role applies to.

This is an administrative subcommand. You must first log on as cluster administrator or consumer administrator before you can issue this subcommand.

-u *user_account*

Specifies the user account to assign the role to. The user account specified must already exist prior to issuing this command.

-r role

Specifies the role to assign. Specify one of the following:

- CLUSTER_ADMIN
- CONSUMER_ADMIN
- CONSUMER_USER

Specify CLUSTER_ADMIN to assign a user account the role of cluster administrator, with administrative authority for all consumers in the cluster. You do not need to specify a path.

Specify CONSUMER_ADMIN to assign a user account the role of consumer administrator for the specified consumer. You must specify the full path to the consumer name over which this user account should have administrative authority.

Specify CONSUMER_USER to assign a user account the role of consumer user. This role has no administrative authority, but is authorized to use resources allocated to the specified consumer. You must specify the full path to the consumer name when specifying this role.

-p consumer_name

Specifies the consumer for which this user is assigned the specified role.

Examples:

The following example assigns George Smith the role of cluster administrator for all the consumers:

```
egoadmin@egosh> user assignrole -u gsmith -r CLUSTER_ADMIN
```

The following example assigns Karen Dayton the role of consumer administrator for the UAT consumer and all of its descendants:

```
egoadmin@egosh> user assignrole -u kdayton -r CONSUMER_ADMIN -p /UAT
```

user delete -u *user_account*

Deletes a user account from the EGO user database.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

-u user_account

Specifies the name of the user account to be deleted.

user list [-l]

Displays all user accounts in the EGO user database and the values specified for phone, email, and description.

-l

Provides the same information with a longer name field, if some are truncated when -l is not specified.

user logoff

Logs off the current user account from EGO. Logging off does not close the interactive command interface session but does prevent the user from issuing administrative subcommands.

user logon -u *user_account* -x *password*

Initiates the log on sequence to EGO, prompting for user account and password.

Note:

You are automatically logged off of EGO after 8 hours. To perform another administrative command after expiry, you are required to log on again.
The logon expiry time is not configurable.

-u *user_account*

Specifies the EGO user account to use to log on.

-x *password*

Specifies the password to use to authenticate the log on sequence.

user modify -u *user_account* [-x *password*] [-e *email*] [-t *telephone*] [-d *description*]

Changes user account values to those specified for a user account defined in the EGO user database.

This is an administrative subcommand. You must first log on as cluster administrator before you can issue this subcommand.

-u *user_account*

Specifies the name of the user account to modify. You cannot modify the name itself.

-x *password*

Specifies the new password to be used to authenticate the user when this user account is accessed.

Specify one to sixteen alphanumeric or special characters. Certain special characters such as greater than (>) and less than (<) are not valid when used with the -x option. In such cases, enter the new password at the prompt after issuing the `user modify` command instead of using the -x option.

-e *email*

Specifies a new email address of the user to whom this account belongs.

Specify up to 64 alphanumeric or special characters, except greater than (>), less than (<), ampersand (&), or control characters (Ctrl + key).

-t *telephone*

Specifies the telephone number of the user to whom this account belongs.

Specify up to 20 numbers and spaces.

-d *description*

Specifies any additional information about the user account or the user to whom this account belongs.

Specify up to 200 alphanumeric or special characters, except control characters (Ctrl + key). Enclose description in quotation marks if there are spaces within it.

user roles4user -u *user_account*

Lists the roles assigned to a user account.

-u user_account

Specifies the user account for which to list the roles.

user users4role -r *role* [-p *consumer_name*]

Lists all user accounts in the EGO user database that have the specified role. For consumer administrators, this command also lists the consumer this user can administer. For consumer users, this command also lists the consumer to which the user has access. For users with custom roles, the consumer is not listed.

-r role

Specifies the role to list all users for. Specify one of the following predefined roles or a custom role:

- CLUSTER_ADMIN
- CONSUMER_ADMIN
- CONSUMER_USER

-p consumer_name

Lists all users' accounts and their roles for the specified consumer. If you specified the role CLUSTER_ADMIN, a consumer_name is not needed. If you specified either of the other two predefined roles, a consumer_name is required. The consumer_name option does not apply to custom roles.

user unassignrole -u *user_account* -r *role* [-p *consumer_name*]

Removes the specified role from the specified user account. Optionally, specifies the consumer to which this action applies or removes this role from all descendants of the specified consumer.

This is an administrative subcommand. You must first log on as cluster administrator or consumer administrator before you can issue this subcommand.

-u user_account

Specifies the user account to remove the role from.

-r role

Specifies the role to remove. Specify one of the following:

- CLUSTER_ADMIN
- CONSUMER_ADMIN
- CONSUMER_USER

-p consumer_name

pecifies the consumer for which this role is removed from the user account.

user view [*user_account* ...]

Displays a list of EGO user accounts.

user_account

Specifies the name of the specific user account(s) to view.

egoshutdown

Shuts down a cluster.

Synopsis

egoshutdown.sh

egoshutdown.bat

Description

Stops all EGO components (EGO daemons or processes such as `l i m`, `pem`, `vemkd`, `egosc`), stops the Platform Management Console, and, if applicable, stops various components belonging to Platform components that might be running on EGO (for example, the Platform Symphony Session Director).

Use this command when you want to shut down a cluster.

This is an administrative command. You must be a cluster administrator to issue this command. On UNIX, you must also be logged on with `root` permissions to issue this command; on Windows, you must be logged on as the OS account administrator, `egoadmi n`.

You can issue this command from any host in the cluster.

egostartup

Starts all EGO components of a cluster.

Synopsis

egostartup.sh

egostartup.bat

Description

Starts all EGO components in a cluster.

This is an administrative command. On UNIX, you must also be logged on with `root` permissions to issue this command; on Windows, you must be logged on as the OS account administrator, `egoadmin`.

You cannot issue this command from a compute host unless the master host is up and running. To use this command on UNIX, you must have `root` permission on each host and have `rsh` configured for your account. You may need to add an entry for the local host in the `.rhosts` file for `root`.

You cannot start EGO on UNIX hosts from a Windows host, or vice versa.

pversions

Displays the version information for Platform products installed on a Windows host. This is a Windows command only; this command is not recognized on UNIX systems.

Synopsis

pversions [*product_name*]

pversions -h

pversions -V

Description

Displays the version and patch level of a Platform product installed on a Windows host, and the list of patches installed.

Options

product_name

Specify the Platform product or component for which you want version information. For example, specify **EGO** to see version information about EGO.

If you have a Platform product installed and running on EGO (for example, Platform Symphony or Platform LSF), you would specify the appropriate product name to see version information for each (for example, specify **Symphony** or **LSF**).

-h

Prints command usage to `stderr` and exits from the software.

-V

Prints product version to `stderr` and exits.

rfa

Transfers files between hosts.

Synopsis

rfa *subcommand* [*options*]

rfa -h

rfa -V

Description

The **rfa** subcommands are useful when transferring files between hosts during installation or package upgrades. Use this command to list files on a specified host, copy files to and from other hosts, or remove files from a host.

You must have permission to access specified directories. You must have appropriate read and/or write privileges on specified hosts.

If you are using a firewall, you need to set the environment variable **EGO_CLIENT_ADDR** on the client to configure the port range for the firewall. It is required to set this variable on the client machine before running the **rfa** command.

Note:

In all cases, if no authentication information is provided (for example, **user_name/password**, **credential**), then existing credentials are automatically used. If there are no existing credentials, you are prompted for a username and password.

-h

Outputs command usage and exits.

-V

Outputs product version and exits.

Subcommand synopsis

list -t *host_name* **-s** *remote_dir* **-p** *consumer_name* [**-c** *credential* | **-u** *user_name* **-x** *password*]

get -t *host_name* **-d** *local_destination_file* **-s** *remote_source_file* **-p** *consumer_name* [**-c** *credential* | **-u** *user_name* **-x** *password*]

put -t *host_name* **-d** *remote_destination_file* **-s** *local_source_file* **-p** *consumer_name* [**-c** *credential* | **-u** *user_name* **-x** *password*]

remove -t *host_name* **-s** *source_file* **-p** *consumer_name* [**-c** *credential* | **-u** *user_name* **-x** *password*] [**-d**]

list -t *host_name* **-s** *remote_dir* **-p** *consumer_name* [**-c** *credential* | **-u** *user_name* **-x** *password*]

Lists files on a specified host.

-t *host_name*

Name of host you want to list files for.

-s remote_dir

Absolute path to the remote directory from which you want to list files.

-p consumer_name

Name of the consumer requesting the resource allocation (which is required to run the `list` command).

-c credential

(Optional.) Authorization ID provided from current EGO logon. Credentials stored here:

- Linux
 - Directory: `/tmp`
 - Parameter: `secegcc_uid`
- Windows:
 - Directory: `%TEMP%\secegcc_osUserName` (where `%TEMP%` is the `TEMP` environment variable)
 - Parameter: `secegcc_osUserName`

Note:

Use the escape character (`\`) before each double quoted character in the credential. For example: **Admin\2007-01-12T02:36:44Z**
\fbmTKOzDeUs1RtU
+gXKJW8PYSYZZCgHXrjciZnBToGSQC/
3tUqYXftYMzv4v1ciJ71wP+wZNeJ6cRxTmzQhOLA==
\lwN0XXwwRXalgM5KlieZbw==

Stored credentials expire after a certain length of time. If you do not specify an authorization ID, you need to specify a user name and password (see `-u` and `-x` options).

-u user_name

(Optional.) Fully-qualified name of the EGO user with associated permissions for the consumer requesting the resource allocation.

-x password

(Optional.) Password to register for the EGO user with associated permissions for the consumer requesting the resource allocation.

get -t host_name -d local_destination_file -s remote_source_file -p consumer_name [-c credential | -u user_name -x password]

Copies a file from a specified host to a local destination.

-t host_name

Name of host you want to copy a file from.

-d local_destination_file

Name of the local destination file, including its directory location, you want to copy to.
Directory path must be absolute.

-s remote_source_file

Name of the remote source file, including its directory location, you want to copy.
Directory path must be absolute.

-p consumer_name

Name of the consumer requesting the resource allocation (which is required to run the
get command).

-c credential

(Optional.) Authorization ID provided from current EGO logon. Credentials stored
here:

- Linux
 - Directory: /tmp
 - Parameter: *secegcc_uid*
- Windows:
 - Directory: %TEMP%\secegcc_*osUserName* (where %TEMP% is the TEMP environment variable)
 - Parameter: *secegcc_osUserName*

Note:

Use the escape character (\) before each double quoted character
in the credential. For example: **Admin\"2007-01-12T02:36:44Z**

\'fbmTKOzDeUs1RtU

+gXKJW8PYSYZZCgHXrjciZnBToGSQC/

3tUqYXftYMzv4v1ciJ71wP+wZNeJ6cRxTmzQhOLA==

\'lwN0XXwwRXalgM5KlieZbw==

Stored credentials expire after a certain length of time. If you do not specify an
authorization ID, you need to specify a user name and password (see **-u** and **-x** options).

-u user_name

(Optional.) Fully-qualified name of the EGO user with associated permissions for the
consumer requesting the resource allocation.

-x password

(Optional.) Password to register for the EGO user with associated permissions for the
consumer requesting the resource allocation.


```
put -t host_name -d remote_destination_file -s
local_source_file -p consumer_name [-c credential |
-u user_name -x password]
```

Copies a file to a specified host from a local location.

-t *host_name*

Name of host you want to copy a file to.

-d *remote_destination_file*

Name of the remote destination file, including its directory location, you want to copy to. Directory path must be absolute.

-s *local_source_file*

Name of the local source file, including its directory location, you want to copy. Directory path must be absolute.

-p *consumer_name*

Name of the consumer requesting the resource allocation (which is required to run the put command).

-c *credential*

(Optional.) Authorization ID provided from current EGO logon. Credentials stored here:

- Linux
 - Directory: /tmp
 - Parameter: secegooc_user_id
- Windows:
 - Directory: C:\Windows\Temp
 - Parameter: egosec_user_id

Note:

Use the escape character (\) before each double quoted character in the credential. For example: **Admin\"2007-01-12T02:36:44Z
 \fbmTKOzDeUs1RtU
 +gXKJW8PYSYZZCgHXrjciZnBToGSQC/
 3tUqYXftYMzv4v1ciJ71wP+wZNeJ6cRxTmzQhOLA==
 \lwN0XXwwRXalgM5KlieZbw==**

Stored credentials expire after a certain length of time. If you do not specify an authorization ID, you need to specify a user name and password (see -u and -x options).

-u *user_name*

(Optional.) Fully-qualified name of the EGO user with associated permissions for the consumer requesting the resource allocation.

-x password

(Optional.) Password to register for the EGO user with associated permissions for the consumer requesting the resource allocation.

```
remove -t host_name -s source_file -p
consumer_name [-c credential | -u user_name -x
password] [-d ]
```

Removes a file from a specified host.

-t host_name

Name of host you want to remove a file from.

-s source_file

Name of the source file, including its directory location, you are removing. Directory path must be absolute.

-p consumer_name

Name of the consumer requesting the resource allocation (which is required to run the remove command).

-c credential

(Optional.) Authorization ID provided from current EGO logon. Credentials stored here:

- Linux
 - Directory: /tmp
 - Parameter: *secegooc_uid*
- Windows:
 - Directory: %TEMP%\secegooc_*osUserName* (where %TEMP% is the TEMP environment variable)
 - Parameter: *secegooc_osUserName*

Note:

Use the escape character (\) before each double quoted character in the credential. For example: **Admin\"2007-01-12T02:36:44Z
 \"fbmTKOzDeUs1RtU
 +gXKJW8PYSYZZCgHXrjciZnBToGSQC/
 3tUqYXftYMzv4v1ciJ71wP+wZNeJ6cRxTmzQhOLA==
 \"lwN0XXwwRXalgM5KlieZbw==**

Stored credentials expire after a certain length of time. If you do not specify an authorization ID, you need to specify a user name and password (see -u and -x options).

-u user_name

(Optional.) Fully-qualified name of the EGO user with associated permissions for the consumer requesting the resource allocation.

-x password

(Optional.) Password to register for the EGO user with associated permissions for the consumer requesting the resource allocation.

-d

(Optional.) The -d flag indicates that the source file specified in the remove command is a directory. This command removes the specified directory and all its contents.

rsdeploy

Deploys and removes packages, such as middleware, upgrade, file, or binary packages.

You must be a cluster administrator to run this command.

Synopsis

rsdeploy *subcommand* [*options*]

rsdeploy -h

rsdeploy -V

Description

Use **rsdeploy** command to remotely deploy packages to a specified host or group of hosts, view the status of current deployment or details about past deployments, cancel a deployment, or remotely uninstall a package.

Restriction:

File package deployment is not intended to work on NFS installations.

-h

Outputs command usage and exits.

-V

Outputs product version and exits.

Subcommand synopsis

add *package_name* **-p** *package_file* **[-o** *os_type* **]** **[-n]** **[-f]** **-u** *user_name* **-x** *password*

cancel *package_name* **-u** *user_name* **-x** *password*

install *package_name* **[-c** *consumer_name* **-r** *resource_group* **]** **[-r** *resource_group* **]** **[-t** *host_name* **]** **[-f]** **-u** *user_name* **-x** *password*

remove *package_name* **[-o** *os_type* **]** **-u** *user_name* **-x** *password*

status *package_name* **[-s** *all* | *allocating* | *waiting* | *active* | *done* | *error* | *cancelled* **]** **-u** *user_name* **-x** *password*

uninstall *package_name* **[-c** *consumer_name* **-r** *resource_group* **]** **[-r** *resource_group* **]** **[-t** *host_name* **]** **-u** *user_name* **-x** *password*

view **-u** *user_name* **-x** *password*

add *package_name* **-p** *package_file* **[-o** *os_type* **]** **[-n]** **[-f]** **-u** *user_name* **-x** *password*

Adds the package to the repository server.

package_name

Specifies the package name.

The package name is used when running other `rsdeploy` commands. Enter a meaningful name, as this name, not the file name, identifies the package in the repository service.

The package name can contain up to 1024 alphanumeric characters, including the characters “.” and “_”. Spaces and symbols are not allowed.

Note:

A Windows limitation prohibits package names ending with “.”. For example, running the command

```
mkdir a . .
```

only creates the directory “a”.

-p *package_file*

Specifies the package file to upload.

Package types can be one of the following: `tar.Z`, `tar.gz`, `tgz`, `gz`, `taz`, `tar`, `jar`, `tar.zip`, or `zip`.

-o *os_type*

Specifies the operating system type, as matched to the host resource type.

Typical usage is to indicate an OS type. You do not need to indicate an OS type for homogeneous clusters (where all hosts in the cluster are either Windows hosts, or all hosts are running the same Linux kernel).

-n

Indicates that package verification is not required.

By default, all packages are validated to ensure they are acceptable for deployment. Packages provided by Platform Computing do not require verification.

-f

Clears the package status for every host prior to installation.

Use this option to update the package status in cases where a manual change may have been done outside by circumventing the `rsdeploy` command. Ensures cached status is cleared, and the actual status is explicitly discovered. This is important as the package is installed only on those hosts that do not have the status “installed”; therefore, if a host reports an outdated status of “installed”, then the package will not be installed when the command is issued.

-u *user_name*

Specifies the EGO user name of the component to which you are issuing this command.

-x *password*

Specifies the EGO user password of the component to which you are issuing this command.

cancel *package_name* -u *user_name* -x *password*

Cancels the remote deployment. Does not cancel in the middle of a package installation on a host, but stops installation on other hosts awaiting package installation.

package_name

Specifies the package name.

-u *user_name*

Specifies the EGO user name of the component to which you are issuing this command.

-x *password*

Specifies the EGO user password of the component to which you are issuing this command.

install *package_name* [-c *consumer_name* -r *resource_group*] [-r *resource_group*] [-t *host_name*] [-f] -u *user_name* -x *password*

Initiates deployment of the package across specified hosts.

package_name

Specifies the package name.

The name was assigned when the package was first added to the repository server.

-c *consumer_name*

Specifies the consumer used to get an allocation to initiate the activity. The full consumer path is required, and must be preceded by a slash (for example, **/ClusterServices/EGOClusterServices**). The consumer path must be to a leaf consumer.

The consumer needs appropriate privileges/permissions to start a activity on the remote host. (Only a cluster administrator has access to all target hosts.)

Note:

If you specify a consumer name in the command, you are required to also specify a resource group.

-r *resource_group*

Specifies the resource group containing all target hosts.

-t *host_name*

Specifies host to which to install the package.

-f

Clears the package status for every host prior to installation.

Use this option to update the package status in cases where a manual change may have been done outside by circumventing the rsdeploy command. Ensures cached status is

cleared, and the actual status is explicitly discovered. This is important as the package is installed only on those hosts that do not have the status "installed"; therefore, if a host reports an outdated status of "installed", then the package will not be installed when the command is issued.

-u *user_name*

Specifies the EGO user name of the component to which you are issuing this command.

-x *password*

Specifies the EGO user password of the component to which you are issuing this command.

remove *package_name* [-o *os_type*] -u *user_name* -x *password*

Removes the package from the repository server.

package_name

Specifies the package name.

-o *os_type*

Specifies the operating system type, as matched to the host resource type.

Typical usage is to indicate an OS type. You do not need to indicate an OS type for homogeneous clusters (where all hosts in the cluster are either Windows hosts, or all hosts are running the same Linux kernel).

-u *user_name*

Specifies the EGO user name of the component to which you are issuing this command.

-x *password*

Specifies the EGO user password of the component to which you are issuing this command.

status *package_name* [-s all | allocating | waiting | active | done | error | cancelled] -u *user_name* -x *password*

Gets the status of deployments, including pending and completed deployments. Lists deployment errors.

package_name

Specifies the package name.

-s

Specifies for filtering criteria for retrieving the status of deployments.

- **all**: Default filter. Retrieves the status on all deployments.
- **allocating**: Retrieves the status on deployments awaiting an allocation from EGO.

- **wait ing**: Retrieves the status on deployments waiting for the remote agent to start.
- **act i ve**: Retrieves the status on deployments with agents started on remote machines.
- **done**: Retrieves the status on deployments that have completed their package installations.
- **error**: Retrieves the status on deployments that have received error messages.
- **cancel led**: Retrieves the status on deployments that were canceled.

-u *user_name*

Specifies the EGO user name of the component to which you are issuing this command.

-x *password*

Specifies the EGO user password of the component to which you are issuing this command.

uninstall *package_name* [-c *consumer_name* -r *resource_group*] [-r *resource_group*] [-t *host_name*] -u *user_name* -x *password*

Uninstalls a package from the hosts.

package_name

Specifies the package name to uninstall.

-c *consumer_name*

Specifies the consumer used to get an allocation to initiate the activity. The full consumer path is required, and must be preceded by a slash (for example, **/ClusterServices/EGOClusterServices**). The consumer path must be to a leaf consumer.

The consumer needs appropriate privileges/permissions to start a activity on the remote host. (Only a cluster administrator has access to all target hosts.)

Note:

If you specify a consumer name in the command, you are required to also specify a resource group.

-r *resource_group*

Specifies the resource group containing all target hosts.

-t *host_name*

Specifies host to which to uninstall the package.

-u *user_name*

Specifies the EGO user name of the component to which you are issuing this command.

-x *password*

Specifies the EGO user password of the component to which you are issuing this command.

`view -u user_name -x password`

Lists the packages in the repository and their creation date.

`-u user_name`

Specifies the EGO user name of the component to which you are issuing this command.

`-x password`

Specifies the EGO user password of the component to which you are issuing this command.

soamcontrol

Controls applications and sessions.

Synopsis

soamcontrol *subcommand* [*options*]

soamcontrol -h

soamcontrol *subcommand* -h

soamcontrol -V

Description

Use the `soamcontrol` command with a subcommand to control applications, sessions, and tasks.

Symphony DE does not verify the optional user name and password. Symphony DE users can control all sessions, applications, and tasks in their environment.

In grid, the user must have the appropriate permissions to control an application, task, or session, as follows:

- Consumer administrators have permission to use this command to control applications, sessions, and tasks in the consumers they manage
- Cluster administrators have permission to use this command to control applications, sessions, and tasks for all consumers
- Consumer users can use this command only to control sessions they submit

-h

Prints command usage to `stdout` and exits.

-V

Prints product version to `stdout` and exits.

Subcommand synopsis

app enable *application_name* [-u *user_name*] [-x *password*]

app disable *application_name* | **all** [-f] [-s] [-u *user_name*] [-x *password*]

session kill *application_name* [:*session_ID* [, *session_ID*]...] [-m *comment*] [-u *user_name*] [-x *password*]

session kill *application_name* [-t *session_tag*] [-m *comment*] [-u *user_name*] [-x *password*]

session suspend *application_name* [:*session_ID* [, *session_ID*]...] [-m *comment*] [-u *user_name*] [-x *password*]

session suspend *application_name* [-t *session_tag*] [-m *comment*] [-u *user_name*] [-x *password*]

session resume *application_name* [:*session_ID* [, *session_ID*]...] [-m *comment*] [-u *user_name*] [-x *password*]

session resume *application_name* [-t *session_tag*] [-m *comment*] [-u *user_name*] [-x *password*]

task kill *application_name:session_ID:task_ID* [, *task_ID*]... [-m *comment*] [-u *user_name*] [-x *password*]

app enable *application_name* [-u *user_name*] [-x *password*]

Enables an application for a consumer with no other enabled applications.

There can only be one enabled application per consumer.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only, the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you have already logged on to Symphony using `soaml ogon`, for this command only the password you specify here overrides the password you entered in `soaml ogon`.

Enable sampleApp application.

```
soamcontrol app enable sampleApp
```

app disable *application_name* | all [-f] [-s] [-u *user_name*] [-x *password*]

Disables a specific application or with the all option, disables all applications to which the specified user has access.

When issued by a cluster administrator, can disable all applications under all consumers in the cluster.

When issued by a consumer administrator, can disable all applications under the consumer this administrator administers.

When an application that has running workload is disabled, all active sessions are immediately killed and all system resources assigned to the application are released.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

all

Disables all applications and kills all workload.

-f

Disables the application and kills application workload without warning if there is active workload for the application.

-s

Disables the application and saves recoverable application workload. Prompts for confirmation to disable the workload. Saved workload is recovered when the application is enabled again.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you have already logged on to Platform Symphony using `soaml ogon`, for this command only the user name you specify here overrides the user name you entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Disable all applications

Disable all applications to which the consumer administrator has access.

soamcontrol app disable all -u consumer_admi -x passwd

Disable application and shutdown workload

Disable application `sampleApp` and shut down any existing workload.

soamcontrol app disable sampleApp -f

session kill *application_name* [:*session_ID* [,*session_ID*]...] [-m *comment*] [-u *user_name*] [-x *password*]

Terminates all sessions of an application or terminates a specific session of an application. The session runtime data and its tasks (such as output data and current state) are no longer available. However, historical data can be retrieved through the `soamvi ew` or the Platform Symphony GUI if session and task history is saved.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

session_ID

Specifies the Platform Symphony -assigned ID of the session as reported by `soamvi ew sessi on`.

-m comment

Specifies a reason why this action was requested. The reason can comprise up to 256 alphanumeric or special characters, with the exception of control (Ctrl + key) and multi-byte characters. The description must be enclosed in double quotes (") if it contains spaces.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Kill all sessions

Kill all sessions of the `sampleApp` application.

```
soamcontrol session kill sampleApp
```

Kill specific sessions

Kill sessions 101 and 102 of the `sampleApp` application.

```
soamcontrol session kill sampleApp:101,102
```

session kill *application_name* [-t *session_tag*] [-m *comment*] [-u *user_name*] [-x *password*]

Terminates all sessions of an application or terminates sessions that match the specified session tag of an application. The session run-time data and its tasks (such as output data and current state) are no longer available. However, historical data can be retrieved through the `soamview` or the Platform Symphony GUI if session and task history is saved.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-t session_tag

Specifies the string that is used for identification purposes to control the session.

-m comment

Specifies a reason why this action was requested. The reason can comprise up to 256 alphanumeric or special characters, with the exception of control (Ctrl + key) and multi-

byte characters. The description must be enclosed in double quotes (" ") if it contains spaces.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Kill all sessions that match the session tag

Kill all sessions of the `sampleApp` application that share the "myTag" session tag.

soamcontrol session kill sampleApp -t myTag

session suspend *application_name* [:*session_ID* [,*session_ID*]...] [-m *comment*] [-u *user_name*] [-x *password*]

Suspends a session. When a session is suspended, running tasks are killed and rescheduled for rerun when the session is resumed. Pending tasks remain pending. CPU slots used by the session are freed.

Clients can continue to create sessions, send tasks, retrieve task output, and query session and task information.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

If specified without one or more session IDs, all sessions of the application are suspended.

session_ID

Specifies the Symphony-assigned ID of the session as reported by `soamview session`.

-m comment

Specifies a reason why this action was requested. The reason can comprise up to 256 alphanumeric or special characters, with the exception of control (Ctrl + key) and multi-byte characters. The description must be enclosed in double quotes (" ") if it contains spaces.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Suspend all sessions

Suspend all sessions of the `sampleApp` application.

soamcontrol session suspend sampleApp

Suspend specific sessions

Suspend sessions 101 and 102 of the `sampleApp` application, giving a reason for the suspension.

soamcontrol session suspend sampleApp:101,102 -m "network congestion"

`session suspend application_name [-t session_tag] [-m comment] [-u user_name] [-x password]`

Suspends all sessions of an application or suspends sessions that match the specified session tag of an application. When a session is suspended, running tasks are killed and rescheduled for rerun when the session is resumed. Pending tasks remain pending. CPU slots used by the session are freed.

Clients can continue to create sessions, send tasks, retrieve task output, and query session and task information.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

If specified without one or more session IDs, all sessions of the application are suspended.

-t session_tag

Specifies the string that is used for identification purposes to control the session.

-m comment

Specifies a reason why this action was requested. The reason can comprise up to 256 alphanumeric or special characters, with the exception of control (Ctrl + key) and multi-byte characters. The description must be enclosed in double quotes (") if it contains spaces.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Suspend all sessions that match the session tag

Suspend all sessions of the `sampleApp` application that share the "myTag" session tag.

soamcontrol session suspend sampleApp -t myTag -m "network congestion"

session resume *application_name* [:*session_ID* [,*session_ID*...]] [-*m comment*] [-*u user_name*] [-*x password*]

Resumes task processing for a specific suspended session or for multiple sessions.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

If specified without one or more session IDs, all sessions of the application are resumed.

session_ID

Specifies the Platform Symphony -assigned ID of the session as reported by `soamview session`.

-m comment

Specifies a reason why this action was requested. The reason can comprise up to 256 alphanumeric or special characters, with the exception of control (Ctrl + key) and multi-byte characters. The description must be enclosed in double quotes (" ") if it contains spaces.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Resume all sessions

Resume all sessions of the sampleApp application.

```
soamcontrol session resume sampleApp
```

Resume a specific session

Resume sessions 101 and 102 of the sampleApp application, giving a reason for the resumption.

```
soamcontrol session resume sampleApp:101,102 -m "network problem resolved"
```

session resume *application_name* [-t *session_tag*] [-m *comment*] [-u *user_name*] [-x *password*]

Resumes all sessions of an application or resumes sessions that match the specified session tag of an application.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

If specified without one or more session IDs, all sessions of the application are resumed.

-t session_tag

Specifies the string that is used for identification purposes to control the session.

-m comment

Specifies a reason why this action was requested. The reason can comprise up to 256 alphanumeric or special characters, with the exception of control (Ctrl + key) and multi-byte characters. The description must be enclosed in double quotes (" ") if it contains spaces.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Resume sessions that match the session tag

Resume all sessions of the sampleApp application that match the "myTag" session tag.


```
soamcontrol session resume sampleApp -t myTag -m "network problem resolved"
```

task kill *application_name:session_ID:task_ID*
[,task_ID...] *[-m comment]* *[-u user_name]* *[-x password]*

Kills one or more specific tasks in a running or pending session.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

session_ID

Specifies the Platform Symphony -assigned ID of the session as reported by `soamview session`.

task_ID

Specifies the Platform Symphony -assigned ID of the task as reported by `soamview task`.

-m comment

Specifies a reason why this action was requested. The reason can comprise up to 256 alphanumeric or special characters, with the exception of control (Ctrl + key) and multi-byte characters. The description must be enclosed in double quotes (") if it contains spaces.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soamlogon`, for this command only the user name specified here overrides the user name entered in `soamlogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soamlogon`, for this command only the password specified here overrides the password entered in `soamlogon`.

Kill specified tasks in a session

Kill tasks 201 and 202 of the sampleApp application in session 101.

```
task kill sampleApp:101:201,202 -m "tasks hanging"
```


soamdeploy

Deploys, removes, and displays information about service packages for consumers.

Synopsis

```
soamdeploy
soamdeploy subcommand [options]
soamdeploy subcommand -h
soamdeploy -h
soamdeploy -V
```

Description

Use the soamdeploy command to deploy, verify, and delete service packages

-h

Prints command usage to stdout and exits.

-V

Prints product version to stdout and exits.

Subcommand synopsis

```
add package_name -p package_file -a application_profile_file | -c consumer_ID [-f] [-n] [-u
user_name] [-x password]
remove package_name -c consumer_ID [-f] [-u user_name] [-x password]
view -c consumer_ID [-u user_name] [-x password]
```

add package_name -p package_file -c consumer_ID [-f] [-n] [-u user_name] [-x password]

Copies the specified package to the central repository.

When you add a package with a new package name to the repository, the package file is stored in the repository. When you add a package with an existing package name:

Note:

If a package is shared among multiple applications, then the following points are applicable for each application.

- The package in the repository is replaced.
- If the package is being used by an enabled application, workload continues to run with the next scheduled task using the updated service package.

When a new session requests the service in the package, the package is deployed and uncompressed on to compute hosts on-demand.

package_name

Name you want to assign to the service package. The service package name must be unique within a consumer, with a maximum of 1024 characters. Valid characters are alphanumeric characters, periods, and underscores. Spaces are not allowed.

-p package_file

Path and file of the service package to deploy.

-a application_profile_file

This command option is applicable to Symphony DE only.

Specify the path and name of the application profile for the application. If any part of the path or file name contains spaces, enclose it in double quotes (" "). Specify the path for the application profile using absolute, relative, or UNC format.

You can specify file paths using absolute or relative paths:

- Absolute—Provide the full path to the file, from the root directory. If you specify an absolute path, all hosts must have the same path. For example:
 - Windows: "%SOAM_HOME%\5.1\samples\CPP\SampleApp\SampleApp.xml"
 - Linux: "\$SOAM_HOME/5.1\samples/CPP/SampleApp/SampleApp.xml"
- Relative—Provide the path to the file relative to the Symphony work directory, %SOAM_HOME%\work in Windows and \$SOAM_HOME/work in Linux. For example:
 - Windows: "..\5.1\samples\CPP\SampleApp\SampleApp.xml"
 - Linux: "../5.1/samples/CPP/SampleApp/SampleApp.xml"

-c consumer_ID

Consumer for which to deploy the service packages. Only applications registered to this consumer and below in the consumer tree can use the service packages.

Enclose the consumer ID in double quotes (" ") if it contains spaces.

If a service package is deployed to the root consumer ("/"), the package is shared by the leaf consumers.

Note:

For Symphony DE, -a application profile file name can be used instead of -c consumer_ID.

-f

Forces package deployment without prompting. Note that if you use this option and the package was previously deployed, any running workload is terminated without prompting. Use this option when you are issuing `soamdeploy add` from within a script, and do not want the script to stop running to respond to prompts.

-n

Turns off package verification during deployment. Use this option if you are deploying the same package repeatedly and you know it is valid, and deployment takes a long time because of the verification process.

-u user_name

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Examples: Windows

Deploy a service package

Deploy the service package for consumer /SampleApplications/SOASamples.

```
soamdeploy add Mypackage -p sampleService.zip -c /SampleApplications/SOASamples
```

Update a service package

Update and replace the service package `sampleService.zip`.

```
soamdeploy add Mypackage -p sampleService.zip -c /SampleApplications/SOASamples
```

Examples: Linux

Deploy a service package

Deploy the service package for consumer /SampleApplications/SOASamples.

```
soamdeploy add Mypackage -p sampleService.tar.gz -c /SampleApplications/SOASamples
```

Update a service package

Update and replace the service package

```
soamdeploy add Mypackage -p sampleService.tar.gz -c /SampleApplications/SOASamples
```

remove package_name -c consumer_ID [-f] [-u user_name] [-x password]

Deletes the specified service package from the central repository under the specified consumer.

Attention:

The package is removed only from the central repository and is not removed from the remote deploy directory until another package is uploaded to the same consumer on that host.

You cannot remove a package if there are registered applications using the package. Unregister the application(s) with `soamunreg` before attempting to remove the package.

Note:

If your application no longer needs to reference the service package and you do not want to unregister it, you can remove reference to the package in the application profile and update the application profile with the `soamreg` command.

package_name

package_name

Name you assigned to the service package during deployment.

-c consumer_ID

Consumer from which to remove the service package.

Enclose the consumer ID in double quotes (" ") if it contains spaces.

-f

Forces package removal without prompting. Use this option when you are issuing `soamdeploy remove` from within a script, and do not want the script to stop running to respond to prompts.

-u user_name

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Remove a service package

Remove the service package for consumer `/SampleApplications/SOASamples`.

```
soamdeploy remove Mypackage -c /SampleApplications/SOASamples
```

`view -c consumer_ID [-u user_name] [-x password]`

Lists all deployed service packages for the specified consumer.

-c consumer_ID

Displays a list of all deployed service packages for the specified consumer. Enclose the consumer ID in double quotes (" ") if it contains spaces.

Note:

For Symphony DE, -c consumer_ID is optional.

-u user _name

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

View deployed packages for consumer /SampleApplications/SOASamples

```
soamdeploy view -c /SampleApplications/SOASamples
```


soamlog

Dynamically changes the log level for Symphony components.

Synopsis

soamlog *subcommand* [*options*]

soamlog -h

soamlog -V

Description

Use `soamlog` to dynamically change the level of detail logged for messages and errors. The log-level change becomes effective immediately and remains in effect until the host is restarted or the affected process or application is closed.

Symphony component log files are written to the `logs` directory in `SOAM_HOME` on the host where the process runs.

When the client path is in the current user's path environment, application logs are written to the directory where the client executable is run. If the client executable path is not in the current user's path environment, application logs are written to the directory where the client executable resides.

Use the `log4j.properties` file on the host on which a Symphony component runs to customize the contents of the log files for that component. The `log4j.properties` file is located in the `conf` directory in `SOAM_HOME`.

In Symphony DE, all users can change any log level.

In Symphony, the cluster or consumer administrator must have permission to use this command. The cluster administrator can change the logging options for the Session Director (`sd`). The consumer administrator can change the log levels for the session manager (`ssm`), service instance manager (`sim`) and workload of their applications.

-h

Prints command usage to `stdout` and exits.

-V

Prints product version to `stdout` and exits.

Subcommand synopsis

sd_all | *sd_logger* -l *log_level* [-u *user_name*] [-x *password*]

ssm_all | *ssm_logger* *application_name* -l *log_level* [-u *user_name*] [-x *password*]

sim_all | *sim_logger* *application_name* -l *log_level* [-u *user_name*] [-x *password*]

soamdeploy_all | *soamdeploy_logger* *application_name* -l *log_level* [-u *user_name*] [-x *password*]

workload *application_name* -l *log_level* [-u *user_name*] [-x *password*]

sd_all | sd_logger -l log_level [-u user_name] [-x password]

sd_all

For activity relating to the Session Director, changes the log level for all logger classes in the `sd.log4j.properties` file, including those without the `sd` prefix.

sd_logger

Specifies the name of the `log4j.logger` you want to change the log level of. For example:

- **SD**—Changes the log level for all logger classes in `sd.log4j.properties`, that are prefixed with `sd`.

Refer to the appropriate `log4j.properties` file (in `$SOAM_HOME/conf` on Linux, `%SOAM_HOME%\conf` on Windows) for the names of all related loggers. If an invalid logger name is specified, the component log levels are not changed.

-l log_level

Specifies one of the following log levels (listed in order of increasing detail):

Level	Description
LOG_INFO	Log all informational messages and more serious messages.
LOG_WARN	Log only those messages that are warnings or more serious messages. This is the default level of debug information.
LOG_ERROR	Log only those messages that indicate error conditions.
LOG_FATAL	Log only those messages in which the system is unusable.
LOG_DEBUG	Log all debug-level messages.
LOG_ALL	Log all available messages.

-u user_name

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Change the log level of the session director

Change the log level of all session director loggers to `LOG_ERROR`.

soamlog sd_all -l LOG_ERROR

ssm_all | ssm_logger application_name -l log_level [-u user_name] [-x password]

ssm_all

For activity relating to the specified application and the related session manager, changes the log level of all logger classes in the `ssm.log4j.properties` file, including those without the `ssm` prefix.

ssm_logger

Specifies the name of the `log4j.logger` you want to change the log level of. For example:

- `ssm`—Changes the log level of all logger classes in the `ssm.log4j.properties` file, that are prefixed with `ssm`.
- `ssm.smbbackend`—Changes the log level of all logger classes in the `ssm.log4j.properties` file, that are prefixed with `backend`.

Refer to the appropriate `log4j.properties` file (in `$SOAM_HOME/conf` on Linux, `%SOAM_HOME%\conf` on Windows) for the names of all related loggers. If an invalid logger name is specified, the component log levels are not changed.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

This application must be registered. Log levels of enabled applications can be changed.

-l log_level

Specifies one of the following log levels (listed in order of increasing detail):

Level	Description
LOG_INFO	Log all informational messages and more serious messages.
LOG_WARN	Log only those messages that are warnings or more serious messages. This is the default level of debug information.
LOG_ERROR	Log only those messages that indicate error conditions.
LOG_FATAL	Log only those messages in which the system is unusable.
LOG_DEBUG	Log all debug-level messages.
LOG_ALL	Log all available messages.

-u user_name

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using `soamlogon`, for this command only the user name specified here overrides the user name entered in `soamlogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soamlogon`, for this command only the password specified here overrides the password entered in `soamlogon`.

Change the log level of the session manager backend loggers

Change the log level of the ssm backend loggers for the sampleApp application to LOG_FATAL.

soamlog ssm.ssmbackend sampleApp -l LOG_FATAL

sim_all | sim_logger application_name -l log_level [-u user_name] [-x password]

sim_all

For activity relating to the specified application and the service instance manager, changes the log level for all logger classes in the `sim.log4j.properties` file, including those without the `sim` prefix.

sim_logger

Specifies the name of the `log4j.logger` for which you want to change the log level. For example:

- **sim**—Changes the log level for all logger classes in the `sim.log4j.properties` file, that are prefixed with `sim`.
- **sim.backend**—Changes the log level for all logger classes in the `sim.log4j.properties` file, that are prefixed with `backend`.

Refer to the appropriate `log4j.properties` file (in `$SOAM_HOME/conf` on Linux, `%SOAM_HOME%\conf` on Windows) for the names of all related loggers. If you specify an invalid logger name, `soamlog` does not change the log level for any component.

If you specify an invalid logger name, `soamlog` does not change the log level for any component.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-l log_level

Specifies one of the following log levels (listed in order of increasing detail):

Level	Description
LOG_INFO	Log all informational messages and more serious messages.
LOG_WARN	Log only those messages that are warnings or more serious messages. This is the default level of debug information.
LOG_ERROR	Log only those messages that indicate error conditions.

Level	Description
LOG_FATAL	Log only those messages in which the system is unusable.
LOG_DEBUG	Log all debug-level messages.
LOG_ALL	Log all available messages.

-u user_name

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Change the log level of the service instance manager for an application

Change the log level of the sim backend loggers for the sampleApp application to LOG_WARN.

soamlog sim.backend sampleApp -l LOG_WARN

soamdeploy_all | soamdeploy_logger application_name -l log_level [-u user_name] [- x password]

soamdeploy_all

For activity relating to `soamdeploy` that are started by SIM, changes the log level for all logger classes in the `soamdeploy.log4j.properties` file, including those without the `soamdeploy` prefix.

soamdeploy_logger

Specifies the name of the `log4j.logger` for which you want to change the log level. For example:

- `soamdeploy`—Changes the log level for all logger classes in the `soamdeploy.log4j.properties` file, that are prefixed with `soamdeploy`.

Refer to the appropriate `log4j.properties` file (in `SSOAM_HOME/conf` on Linux, `%SOAM_HOME%\conf` on Windows) for the names of all related loggers. If you specify an invalid logger name, `soaml og` does not change the log level for any component.

If you specify an invalid logger name, `soaml og` does not change the log level for any component.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-l log_level

Specifies one of the following log levels (listed in order of increasing detail):

Level	Description
LOG_INFO	Log all informational messages and more serious messages.
LOG_WARN	Log only those messages that are warnings or more serious messages. This is the default level of debug information.
LOG_ERROR	Log only those messages that indicate error conditions.
LOG_FATAL	Log only those messages in which the system is unusable.
LOG_DEBUG	Log all debug-level messages.
LOG_ALL	Log all available messages.

-u user_name

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Change the log level of soamdeploy for an application

Change the log level of all soamdeploy loggers to LOG_ERROR.

soamlog soamdeploy_all sampleApp -l LOG_ERROR

workload application_name -l log_level [-u user_name] [-x password]

For activity relating to the specified application, changes the log level of workload-related log entries in the session manager (ssm), session director (sd) and service instance manager (sim) log files.

Change the workload log level for an application

Change the log level of the sampleApp application

soamlog workload sampleApp -l LOG_DEBUG

soamlogoff

Ends the login session with Platform Symphony .

Synopsis

soamlogoff

soamlogoff -h

soamlogoff -V

Description

In Symphony, use `soaml ogoff` to end a Symphony user session.

It is not necessary to log on or log off from Symphony DE.

Once logged off, the user must log on again to issue Symphony commands or must enter a name and password as options to the Symphony command.

It is not necessary to log off to issue commands under a different user account. Specifying a user name and password on the command line overrides the user name and password specified with `soaml ogon`.

-h

Prints command usage to `stdout` and exits.

-V

Prints product version to `stdout` and exits.

soamlogon

Logs the user on to Platform Symphony for a specific time period.

Synopsis

soamlogon [-u *user_name*] [-x *password*] [-t *time-to-live*]

soamlogon -h

soamlogon -V

Description

Use soaml ogon to log on to Symphony for a specific time period.

You do not need to log on to use Symphony DE.

When soaml ogon is used without any options, the system prompts for a user name and password. Symphony does not prompt for an expiry time; it uses the default command session length, which is 8 hours.

Once the command session time-to-live expires, the command session is terminated.

The user name and password specified using soaml ogon apply to all commands issued during this command session, unless they are overridden on the command line using the user name (-u) and password (-x) command options.

When a command is run with the -u and -x options, the system checks the user name and password to ensure the user has the required authority to run that command. The user name and password entered as options to a command apply only to that command instance.

-h

Prints command usage to stdout and exits.

-V

Prints product version to stdout and exits.

Options

-t time-to-live

Default=480 minutes (8 hours). Specifies the command session duration in minutes. When the time expires, you are automatically logged off the system. To run more commands, you must log on again or provide the user name and password on the command line as command options.

-u user _name

Specifies the name of the user to connect to Symphony for this command.

-x password

Specifies the user password to connect to Symphony for this command.

soammod

Modifies session priority to enable high priority workload to finish faster.

Synopsis

soammod *subcommand* [options]

soammod session -h

soammod session -V

Description

Use **soammod session** to change the priority of one session, all sessions, or sessions that match the session tag of an application. Symphony increases or decreases the amount of resources assigned to a session depending on session priority.

Any priority changes made are applied at a time determined by the `Consumer > sessionSchedulingInterval` attribute of the application profile. When Symphony next checks the workload to determine if it must reassign resources, it examines any session priority changes and reallocates resources accordingly.

Changing the priority of a session does not affect running tasks. The system waits for the task to finish before reassigning the resource.

Platform Symphony DE does not verify the optional user name and password. Platform Symphony DE users can control all sessions and applications in their environment.

In the Symphony environment:

- Consumer users can only change sessions they submitted.
- Consumer administrators can change sessions of all applications under that consumer.
- Cluster administrators can change all sessions under all consumers in the cluster.

-h

Prints command usage to `stdout` and exits.

-V

Prints product version to `stdout` and exits.

Subcommand synopsis

session *application_name* [-u *user_name*] [-x *password*] -p *priority_number*

session *application_name:session_ID* [-u *user_name*] [-x *password*] -p *priority_number*

session *application_name* [-u *user_name*] [-x *password*] -p *priority_number* -t *session_tag* [-f]

session *application_name* [-u *user_name*] [-x *password*] -r "priority" -t *session_tag*

session *application_name* [-u *user_name*] [-x *password*] -p *priority_number*

Changes the priority of all sessions for the application.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-p priority_number

Specifies the priority number to assign to all sessions of the specified application. The priority number range is from 1 to 10,000, where priority 1 is the lowest priority.

-u user_name

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Change the priority of all sessions

Change the priority of all sampleApp sessions to 10.

soammod session sampleApp -p 10

session application_name:session_ID [-u user_name] [-x password] -p priority_number

Changes the priority of a specific session for the application.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

session_ID

Specifies the Platform Symphony-assigned ID of the session as reported by `soamview session`.

-p priority_number

Specifies the priority number to assign to the specified session of the specified application. The priority number range is from 1 to 10,000, where priority 1 is the lowest priority.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Change the priority of a specific session.

Change the priority of the specific session to 100.

soammod session sampleApp:101 -p 100

session application_name [-u user_name] [-x password] -p priority_number -t session_tag [-f]

Changes the priority of all sessions for the application that share the specified session tag.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-p priority_number

Specifies the priority number to assign to all current sessions of the specified application that share the given session tag. The priority number range is from 1 to 10,000, where priority 1 is the lowest priority.

-t session_tag

Specifies the string that is used for identification purposes to control the session.

-f

Specifies that future sessions created with the given session tag will inherit the specified priority, rather than get their priority from the application profile. The `-f` flag can only be used when the `-t session_tag` option and the `-p priority` option are both specified. If the `-f` flag is not specified, the priority change applies only to current sessions that share the given session tag.

-u user_name

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Change the priority of current and future sessions

Change the priority of all current and future sampleApp sessions that share the "myTag" session tag to 10.

```
soammod session sampleApp -p 10 -t myTag -f
```

session *application_name* [-u *user_name*] [-x *password*] -r "*priority*" -t *session_tag*

Resets the priority of all future sessions for the application that share the specified session tag.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-r "priority"

Resets the priority change that was previously specified with the -f flag for future sessions. Future sessions that are created with the matching session tag will use the priority from the application profile.

-t session_tag

Specifies the string that is used for identification purposes to control the session.

-u user_name

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using soaml ogon, for this command only the user name specified here overrides the user name entered in soaml ogon.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using soaml ogon, for this command only the password specified here overrides the password entered in soaml ogon.

Reset the priority of future sessions

Reset the priority of future sampleApp sessions created with the "myTag" session tag. Future sessions created with the "myTag" session tag will get their priority from the application profile.

```
soammod session sampleApp -t myTag -r "priority"
```


soamreg

Registers an application or updates the application profile of a registered application.

Synopsis

```
soamreg application_profile_file [-d][-f] [-u user_name] [-x password]
soamreg fragment_profile_file -p [-f] [-u user_name] [-x password]
soamreg -h
soamreg -V
```

Description

Use soamreg to register a new application or update an application with an existing profile or profile fragment.

Before you register an application in Symphony:

- The consumer specified in the application profile must already exist in the resource group names specified, or in the default resource groups.
- The service packages specified in the application profile or profile fragment must have already been deployed for the consumer

Remember:

If the application profile resourceGroupName attribute for the session manager and service instance manager is not defined, the default compute and management host groups are used. In this case, the consumer must be defined in the ComputeHost and the ManagementHost. Otherwise, the application cannot be enabled after registration.

In Symphony DE, the consumer name is optional and not used.

The table that follows describes Symphony behavior when registering an application, based on existing conditions when soamreg was issued:

	No enabled application in consumer	No enabled application in consumer	Enabled application in consumer	Enabled application in consumer	Enabled application in consumer	Enabled application in consumer
			Application has open or suspended sessions	Application has open or suspended sessions	Application has no open or suspended sessions	Application has no open or suspended sessions
System action resulting from issuing soamreg	Register new application	Update existing application	Register new application	Update existing application	Register new application	Update existing application
Registers and enables the application	✓			✓		✓
Registers the application, and it remains disabled		✓	✓		✓	

System action resulting from issuing <code>soamreg</code>	No enabled application in consumer	No enabled application in consumer	Enabled application in consumer	Enabled application in consumer	Enabled application in consumer	Enabled application in consumer
			Application has open or suspended sessions	Application has open or suspended sessions	Application has no open or suspended sessions	Application has no open or suspended sessions
	Register new application	Update existing application	Register new application	Update existing application	Register new application	Update existing application
Registers the application profile fragment		✓		✓		✓
Terminates all sessions				✓		
Shuts down session manager, releases resources, and starts up a new session manager				✓		✓
Updates the application profile		✓		✓		✓
Dynamic update for application profile		✓		✓		✓

-h

Prints command usage to `stdout` and exits.

-V

Prints product version to `stdout` and exits.

Options

`application_profile_file`

Specify the path and name of the application profile or profile fragment file for the application. If any part of the path or file name contains spaces, enclose it in double quotes (" "). Specify the path for the application profile using absolute, relative, or UNC format.

You can specify file paths using absolute or relative paths:

- **Absolute**—Provide the full path to the file, from the root directory. If you specify an absolute path, all hosts must have the same path. For example:
 - Windows: "%SOAM_HOME%\5.1\samples\CPP\SampleApp\SampleApp.xml"
 - Linux: "\$SOAM_HOME/5.1/samples/ CPP/SampleApp/SampleApp.xml"
- **Relative**—Provide the path to the file relative to the Symphony work directory, %SOAM_HOME%\work in Windows and \$SOAM_HOME/work in Linux. For example:
 - Windows: "..\5.1\samples\CPP\SampleApp\SampleApp.xml"

- Linux: `"../5.1/samples/CPP/SampleApp/SampleApp.xml"`

fragments_profile_file

Register a profile fragment file that contains either a new session type or a new service section, or both. When you *add* a new session type or Service section, the system does not terminate running workload. When you *update* an existing Service section or session type, the system only terminates running workload associated with the updated service or session type. The add, update, and remove actions are also applicable for multiple session types and services.

You can specify file paths using absolute or relative paths as explained in `application_profile_file`.

-f

Registers a new application without warning. For existing applications, updates the existing application profile and terminates application workload without warning, if there is active workload for the application.

Registers a profile fragment and does not prompt you to confirm the addition or update.

-p

Registers a profile fragment with the new session types and Service sections, or with the updated or removed session types and Service sections. You can refer to the profile fragment templates to create your own profiles. You can find templates in the `SSOAM_HOME/5.1/samples/Templates` directory.

-d

Updates the application profile dynamically. If you use this option with `-f`, then affected workload will be terminated without any warning.

-u user_name

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Register an application

Register an application on Linux:

```
soamreg soam/5.1/samples/CPP/SampleApp/SampleApp.xml
```

Register a profile fragment:

```
soamreg soam/5.1/samples/CPP/SampleApp/SampleAppFragment.xml -p
```

Register an application on Windows:

```
soamreg "\\hostA\share\SampleApp.xml"
```


Register a profile fragment:

```
soamreg "\\hostA\share\SampleAppFragment.xml" -p
```


soamshutdown

In Symphony DE only, immediately shuts down Symphony processes on the local host.

Synopsis

soamshutdown [-all]

soamshutdown -h

soamshutdown -V

Description

In Symphony DE only, use soamshutdown to shut down Symphony processes on the local host or all hosts in the cluster.

The taskCleanupPeriod and suspendGracePeriod for SessionTypes you specify in the application profile do not apply when you issue this command. The soamshutdown command immediately shuts down all Platform Symphony processes.

On a management host, soamshutdown stops the following processes:

- Session director (sd)
- Session manager (ssm)
- Repository service (rs)
- Platform Management Console (webgui) and Java processes
- Platform EGO emulator (start_agent)

On a compute host, soamshutdown stops the following processes:

- Service instance manager (sim)
- Platform EGO emulator (start_agent)

Options

-all

Immediately shuts down Symphony processes on all hosts in the cluster as configured in vem_resource.conf.

-h

Prints command usage to stdout and exits.

-V

Prints product version to stdout and exits.

soamstartup

In Symphony DE only, starts Symphony processes on the local host.

Synopsis

soamstartup

soamstartup -h

soamstartup -V

Description

In Symphony DE, use `soamstartup` to start Symphony software processes on the local host under the authority of the login user.

The processes `soamstartup` starts depend on the host role, management or compute.

- If the host is a management host, as specified `inven_resource.conf`, `soamstartup` starts:
 - Platform EGO emulator (`start_agent`)
 - Session director (`sd`)
 - Repository service (`rs`)
 - Platform Management Console (`webgui`) and Java processes
- If the host is a compute host, `soamstartup` starts:
 - Platform EGO emulator (`start_agent`)
 - For any applications specified to be prestarted in their application profiles, `soamstartup` also starts:
 - Session manager (`ssm`)
 - Service instance manager (`si m`)
- Processes started when the session director receives workload requests or enabled applications are specified to prestart in their application profiles are:
 - Session manager (`ssm`)
 - Service instance manager (`si m`)

Attention:

For Windows users only—If Symphony DE is installed by the administrator, `start_agent` runs as a Windows service. Closing the command shell under which Symphony DE was started or logging off Windows does not stop the `start_agent` service. However, installing Symphony DE as a non-administrative user makes Symphony DE run as a normal Windows process. In this case, if you close the command shell from which `soamstartup` was issued, the `start_agent` process is killed but any processes it started, such as the session director or session manager continue to run and must be shut down using `soamshutdown`. If the user logs off Windows, all DE processes will be stopped.

-h

Prints command usage to `stdout` and exits.

-V

soamstartup

Prints product version to `stdout` and exits.

soamswitch

Windows only. Resets the system environment for the application client: connecting to a DE cluster from the DE installation environment; connecting to a Symphony cluster from the DE installation environment; or connecting to a Symphony cluster using the Symphony installation environment.

Synopsis

soamswitch sym

soamswitch symde

soamswitch sym *Symphony_dir* [-v *version*] [-h]

soamswitch sym *SymphonyDE_dir* [-v *version*] [-h]

soamswitch symde *SymphonyDE_dir* [-v *version*] [-h]

soamswitch info

soamswitch -h **-V**

Description

You must have local administrator privileges on the host to use `soamswitch`.

Resets the system environment for this machine, specifying both the installation environment to use, and the cluster to connect to.

After the `soamswitch` command is issued, environment changes take effect on all subsequent command sessions—the environment in the current command session is not affected.

Options

soamswitch sym

Use this command to reset your environment to connect to the Symphony cluster instead of the DE cluster. Continue to connect from the DE installation environment.

You can only use this command when the current installation environment is set to DE.

Your environment uses the DE libraries, and work submitted from client applications or commands issued runs on the Symphony cluster in subsequent command sessions.

soamswitch symde

Use this command to reset your environment to connect to the DE cluster instead of the Symphony cluster. Continue to connect from the DE installation environment.

You can only use this command when the current installation environment is set to DE.

Your environment uses the DE libraries, and work submitted from client applications or commands issued runs on the DE cluster in subsequent command sessions.

soamswitch sym *Symphony_dir* -v *version*

Use this command to reset your environment to connect to the Symphony cluster from the Symphony installation environment, regardless of your current environment.

You can only use this command if Symphony is installed on this machine.

Your environment uses the Symphony libraries, and work submitted from client applications or commands issued runs on the Symphony cluster in subsequent command sessions.

Symphony_dir

Specify the path to the directory in which Symphony is installed.

Note:

You cannot specify a UNC path. For example, do not specify \\hosta\share\.

-v version

Specifies the version of Symphony to switch to.

Use this option when multiple versions of Symphony are installed on the same machine.

Reset from DE environment and cluster to Symphony grid environment and cluster

```
soamswitch sym C:\EGO -v 5.1
```

soamswitch sym *SymphonyDE_dir* -v *version*

Use this command to reset your environment to connect to the Symphony cluster from the DE installation environment, regardless of your current environment.

You can only use this command if DE is installed on this machine.

Your environment uses the DE libraries, and work submitted from client applications or commands issued runs on the Symphony cluster in subsequent command sessions.

SymphonyDE_dir

Specify the path to the directory in which DE is installed.

Note:

You cannot specify a UNC path. For example, do not specify \\hosta\share\.

-v version

Specifies the version of Symphony DE to switch to.

Use this option when multiple versions of Symphony DE are installed on the same machine.

Reset from Symphony environment to the DE environment while connecting to the Symphony cluster

```
soamswitch sym C:\SymphonyDE\DE5.1
```


soamswitch symde *SymphonyDE_dir* -v *version*

Use this command to reset your environment to connect from the DE installation environment to the DE cluster, regardless of your current environment.

Your environment uses the DE libraries, and work submitted from client applications or commands issued runs on the DE cluster in subsequent command sessions.

SymphonyDE_dir

Specify the path to the directory in which Symphony DE is installed.

Note:

You cannot specify a UNC path. For example, do not specify \\hosta\share\.

-v *version*

Specifies the version of Symphony DE to switch to.

Use this option when multiple versions of Symphony DE are installed on the same machine.

Reset from Symphony grid environment and cluster to the DE environment and cluster

```
soamswitch symde C:\SymphonyDE\DE5.1
```

soamswitch info

The output of the command indicates your current environment and whether you are connecting to DE cluster or a Symphony cluster.

The output of the command indicates whether your installation environment is set to Symphony or DE.

View my current environment

View whether the environment is currently set to connect to a Symphony cluster.

```
soamswitch info
```

-h

Prints command usage to stdout and exits.

-v

Prints product version to stdout and exits.

Related files

The following configuration files specify which cluster to connect to. The paths shown result from using the default installation directories.

- C: \SymphonyDE\DE51\conf\ego. conf—specifies which Symphony cluster to connect to from a DE installation environment
- C: \SymphonyDE\DE51\conf\vem_resource. conf—specifies which DE cluster to connect to from a DE installation environment
- C: \EG0\kernel \conf\ego. conf—specifies which Symphony cluster to connect to from a Symphony installation environment

soamunreg

Unregisters an application and deletes the application profile, preventing more sessions from being started for this application.

Synopsis

```
soamunreg application_name [-f] [-s] [-u user_name] [-x password]
```

```
soamunreg -h | -V
```

Description

Use `soamunreg` to unregister an application.

Unregistering an application does the following:

- Terminates existing sessions and tasks (running and suspended)
- Disables the application
- Releases all resources allocated to the application
- Removes the application profile from the system

After unregistering the application, remove the application files in `$EGO_CONFDIR/. . . /profiles` on Linux, `%EGO_CONFDIR%\ . . . \profiles` on Windows.

You can optionally save historic data associated with the application

-h

Prints command usage to `stdout` and exits.

-V

Prints product version to `stdout` and exits.

Options

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

You can view the application name using `soamview app`.

-f

Forces the action, bypassing prompts to unregister an enabled application with running workload and to save the historical data. This option assumes you want to unregister the application immediately.

-s

Saves the application historical data.

-u user_name

soamunreg

Specifies the name of the user to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Unregister an application

Unregister an application, `sampleApp`:

```
soamunreg sampleApp
```


soamview

Displays information about applications, sessions, and tasks, and displays the application profile.

Synopsis

soamview *subcommand* [*options*]

soamview *subcommand* -h

soamview -h | -V

Description

Use `soamview` to view information about applications, sessions, and tasks, and to view the application profile.

Symphony DE does not verify the optional user name and password. Symphony DE users can control all sessions and applications in their environment.

In Symphony, the information `soamview` displays depends on your user ID authorization, as follows:

- The cluster administrator can view all information.
- The consumer administrator can view information for all applications associated with that consumer. To specify an absolute consumer name that contains spaces, enclose the name in double quotes (" ").
- Users can view information for only those applications their user privileges allow, the applications they submitted.

To view large amounts of information (when output exceeds one screen), pipe or redirect the command output to a file.

The application totals displayed by `soamview` reflect the values of counters that start when the application is enabled or when a new session manager starts for this application (in a failover event). These counters are reset when the application is disabled or the session manager exits. The session and task totals `soamview` displays reflect counters in session and task runtime and historical data.

Note:

The information listed for SI startup failures lists the last five hosts on which a service instance failed to start. If more than five hosts experienced SI startup failures, only the last five hosts are listed.

-h

Prints command usage to `stdout` and exits.

-V

Prints product version to `stdout` and exits.

Subcommand synopsis

app *application_name* [-l | -p] [-u *user_name*] [-x *password*]

app [-c *consumer_ID*] [-s "all|disabled|enabled"] [-u *user_name*] [-x *password*]

package *application_name* [-r *host_name*] [-a] [-s downloading|finish|error|all] [-t *time_length*] [-w] [-u *user_name*] [-x *password*]

resource *application_name* [:*host_name*] [-a] [-g *resource_group*] [-s *occupied|standby|releasing|all*] [-u *user_name*] [-x *password*]

rg *application_name* [-a] [-g *resource_group_filter*] [-s *sessionId*""] [-u *user_name*] [-x *password*]

session *application_name* [-c *create_time_interval*] [-e *end_time_interval*] [-d] [-s "all|open|closed|suspended|aborted"] [-t *session_tag*] [-U *submission_user*] [-S *session_name*] [-n *counter*] [-u *user_name*] [-x *password*]

session *application_name:session_ID* [-l] [-u *user_name*] [-x *password*]

task *application_name:session_ID* [-b *start_time_interval*] [-c *submit_time_interval*] [-d] [-e *end_time_interval*] [-s "task_state"] [-t *task_tag*] [-n *counter*] [-r *host_name*] [-u *user_name*] [-x *password*]

task *application_name:session_ID:task_ID* [-l] [-u *user_name*] [-x *password*]

app *application_name* [-l | -p] [-u *user_name*] [-x *password*]

Displays brief and summary information for a specific application.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-l

Provides detailed information.

-p

Displays the application profile in XML format.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using soaml ogon, for this command only the user name specified here overrides the user name entered in soaml ogon.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using soaml ogon, for this command only, the password specified here overrides the password entered in soaml ogon.

View information for an application

Display summary and brief information for the sampleApp application.

soamview app sampleApp

app [-c *consumer_ID*] [-s "all | disabled | enabled"] [-u *user_name*] [-x *password*]

Displays information about all enabled and disabled applications the user manages.

In Symphony DE, users have access to information on all applications and do not need to supply the application name, consumer ID, user name, or password.

In Platform Symphony, users without cluster or consumer privileges must provide an application name. Users can view information only for applications, sessions, and tasks they submitted.

-c *consumer_ID*

Specifies the consumer ID. The consumer ID is the same as it appears in the application profile. Enclose the consumer ID in double quotes (" ") if it contains spaces. The consumer ID starts with a forward slash.

-s all

Displays information for all applications in all states.

-s enabled

Displays information for all enabled applications.

-s disabled

Displays information for all disabled applications.

-s "enabled | disabled"

Displays information for all enabled and disabled applications. Enclose the states in double quotes (" ") and use | as a separator. For example, -s "enabled|disabled".

-u *user_name*

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using soaml ogon, for this command only the user name specified here overrides the user name entered in soaml ogon.

-x *password*

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using soaml ogon, for this command only the password specified here overrides the password entered in soaml ogon.

Display information for all applications in the consumer

Display summary and brief information for all applications in the /SampleApplications/SOATesting consumer.

soamview app -c /SampleApplications/SOATesting -s all


```
package application_name [-r host_name] [-s
"downloading | finish | error | all"] [-t time_length] [-w]
[-u user_name] [-x password]
```

Displays service package download details.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-r host_name

Displays the package download details of the specified host(s).

-s downloading

Displays the details for packages that are in the process of being downloaded.

-s finish

Displays the details for package downloads that have finished.

-s error

Displays the details for package downloads that have received error messages.

-s all

Displays package download details for the specified application in all states.

-t time_length

Specifies a duration, in seconds, to query package downloads which are either finished or still in progress within the *time_length* until now.

-w

Specifies wider fields for the host name and package name.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Display package download details

Display package download summary with wide fields for the specified application.

soamview package app -w

resource application_name [: *host_name*] [-a] [-g *resource_group*] [-s occupied|standby|releasing|all] [-u *user_name*] [-x *password*]

Displays information about host slots and allocation status of an application.

Note:

If the display shows slots in the occupied state without an associated session, this means that the slots are allocated to the SSM but are not bound to any session.

Slots in the standby state are always displayed without an associated session since these slots are not allocated to the application.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

host_name

Displays the allocation of the specified host.

-a

Displays the hosts and slots allocation information at the session level.

-g resource_group

Specifies a resource group to filter the results.

-s slot_state

Specifies that information for slots in the given state is required.

-s all

Specifies that information for slots in all states is required.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using soaml ogon, for this command only the user name specified here overrides the user name entered in soaml ogon.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using soaml ogon, for this command only the password specified here overrides the password entered in soaml ogon.

Display host and slots allocation for an application

Display allocation information of the hosts and slots for the sampleApp application.

Note:

If the -s option is not specified, only allocated hosts and slots are displayed.

soamview resource sampleApp -s all

Display session-level host and slots allocation for an application

Display allocation information for each open session of the sampleApp application.

soamview resource sampleApp -a -s all

rg application_name [-a [-g resource_group_filter] [-s sessionId[""]] [-u user_name] [-x password]

When no options are specified, the command displays slot use and demand information of each resource group specified in the resource group filter for an application.

When options are specified, the command displays slot use and demand information for each session including the resource group filter, assigned slots, and unmet demand. For slots that cannot be used by any session when the SSM holds them due to "exclusive allocation", the command also displays unassigned slots.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-a

Displays the resource group filter, assigned and unassigned slots, and unmet demand at the session level.

-g resource_group_filter

Specifies a resource group to filter the results.

-s sessionId

Specifies a session ID to filter the results.

-s ""

Specifies that you want to filter the result for slots that are not allocated to any sessions.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Display session-level slot use and demand information for a specific resource group filter

Display slot use and demand information for each open session that uses the ComputeHosts resource group filter.

soamview rg sampleApp -a -g ComputeHosts

session application_name [-c create_time_interval] [-e end_time_interval] [-d] [-s "all | open | closed | suspended | aborted"] [-t session_tag] [-U submission_user] [-S session_name] [-n counter] [-u user_name] [-x password]

Displays information for sessions in various states. By default, sessions are displayed from newest to oldest. When used without the `-s` option, displays only information for open sessions.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-c create_time_interval

Specifies the interval for the session creation time. If you specify the session creation time interval without an end time, the interval end time is considered the time at which you entered the command.

Specify the time in the format "[yyyy/][mm/][dd/]hh:mm", "[yyyy/][mm/][dd/]hh:mm]". Do not specify spaces in the time interval string. For more specific syntax and examples of time formats, refer to [Time interval format](#) on page 116.

Note:

(If the interval for session creation time is not specified, the default is anytime.)

-e end_time_interval

Specifies the interval for the session end time. If you specify the session end time interval without a session creation time interval, the system retrieves all relevant records prior to the session end time.

Specify the time interval in the format "[yyyy/][mm/][dd/]hh:mm], [[yyyy/][mm/][dd/]hh:mm]". Do not specify spaces in the time interval string. For more specific syntax and examples of time formats, refer to [Time interval format](#) on page 116.

Note:

(If the interval for session end time is not specified, the default is anytime.

-d

Specifies the reverse order for record retrieval starting with the oldest records being displayed first.

-s all

Specifies that information for sessions in all states is required.

-s session_state

Specifies that information for all sessions in one or more of the specified states is required. To specify multiple states, enclose the states in double quotes (" ") and use | as a separator. For example, -s "open|suspended".

Only the first three letters of the state need to be specified.

You can specify an abbreviated form of the session state:

- aborted—abrt
- open—open
- suspended—susp
- closed—clsd

Tip:

The `soamview session -s aborted` command displays information only for sessions that were aborted during the time period specified by `lastingPeriod` in the application profile.

Note:

If the session state is not specified, the default state is open|suspended.

-t session_tag

Specifies the string that is used for identification purposes to query the session.

The session tag supports the use of wildcard characters to represent specific string patterns, as follows:

Note:

Wildcard characters must be surrounded by quotes.

- "*" represents 0 or more characters

For example, to display all sessions in the open state (including sessions with and without tags), enter:

soamview session symping -t "*" -s open

- "?" represents 1 character

For example, to display only sessions in the open state that have tags, enter:

soamview session symping -t "?" -s open

The session tag also supports the use of reserved character "!" to indicate no session tag. For example, to display all sessions in the open state that do not have tags, enter:

soamview session symping -t "!" -s open

Note:

The "!" tag is a standalone reserved character and cannot be used to negate a regular expression.

-U submission_user

Displays information for sessions submitted by the user specified.

-S session_name

Displays information for sessions with the specified session name.

-n counter

Returns the maximum number of records specified.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Display information for sessions submitted by a specific user

Display information for 40 open or suspended sessions for the sampleApp application that were submitted by userabc.

soamview session sampleApp -s "open|susp" -U userabc -n 40

Display information for sessions that match the session tag

Display information for open or closed sessions for the sampleApp application that match the "myTag" session tag.

```
soamview session sampleApp -s "open|closed" -t myTag
```

Display information for sessions with the specified session name

Display information for sessions with the "mySession" session name.

```
soamview session sampleApp -s "open|closed" -S mySession
```

Display information for sessions that match the specified session state during a specified time interval

Display information for sessions that were running during the specified interval. In this example, the system will return all sessions that were open or suspended between 10:00am January 21, 2009 and 2:00pm January 22, 2009.

```
soamview session sampleApp -s "all" -c ,2009/01/22/14:00 -e 2009/01/21/10:00,-
```

Display information for sessions that were created during a specified interval

Display information for all sessions that were created during the specified interval.

```
soamview session sampleApp -s "all" -c 2009/01/21/10:00, 2009/01/23/10:00
```

session application_name:session_ID [-l] [-u user_name] [-x password]

Displays brief and summary information for a specific session of an application.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

session_ID

Specifies the Platform Symphony-assigned ID of the session as reported by `soamview session`.

-l

Provides detailed information.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Display information for a specific session of an application

Display detailed information for a specific session of the sampleApp application.

soamview session sampleApp:101 -l

task application_name: session_ID [-b start_time_interval] [-c submit_time_interval] [-d] [-e end_time_interval] [-r host_name] [-s "task_state"] [-t task_tag] [-n counter] [-u user_name] [-x password]

Displays brief and summary task information for tasks of an application. By default, tasks are displayed from newest to oldest.

application_name

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

session_ID

Specifies the Platform Symphony-assigned ID of the session as reported by `soamview session`.

-b start_time_interval

Specifies the interval for when the task started to run.

Note:

In cases where the task has been restarted, the start time represents the last time the task started to run.

Specify the time interval in the format "[[yyyy/][mm/][dd/]hh:mm], [[yyyy/][mm/][dd/]hh:mm]". Do not specify spaces in the time interval string. For more specific syntax and examples of time formats, refer to [Time interval format](#) on page 116.

-c submit_time_interval

Specifies the interval for the task submit time. If you specify the task submit time interval without an interval end time, the interval end time is considered the time at which you entered the command.

Specify the time interval in the format "[yyyy/]mm/]dd/]hh:mm], [[yyyy/]mm/]dd/]hh:mm]". Do not specify spaces in the time interval string. For more specific syntax and examples of time formats, refer to [Time interval format](#) on page 116.

-d

Specifies the reverse order for record retrieval starting with the oldest records being displayed first.

-e end_time_interval

Specifies the interval for the task end time. If you specify the task end time without a task submit time, the system retrieves all relevant records prior to the task end time.

Specify the time interval in the format "[yyyy/]mm/]dd/]hh:mm], [[yyyy/]mm/]dd/]hh:mm]". Do not specify spaces in the time interval string. For more specific syntax and examples of time formats, refer to [Time interval format](#) on page 116.

-r host_name

Specifies the name of a specific compute host that the tasks were running on.

-s task_state

Displays information for all tasks in the state specified.

You can specify the full task state name or the first three letters of the task state:

- pending
- running
- done
- error
- canceled

-t task_tag

Displays information for all tasks that match the task tag.

The task tag supports the use of wildcard characters to represent specific string patterns, as follows:

Note:

Wildcard characters must be surrounded by quotes.

- "*" represents 0 or more characters

For example, to display all tasks in the done state (including tasks with and without tags), enter:

soamview task symping:901 -t "*" -s done

- "?" represents 1 character

For example, to display only tasks in the done state that have tags, enter:


```
soamview task symping:901 -t "?" -s done
```

The task tag also supports the use of reserved character "!" to indicate no task tag. For example, to display all tasks in the done state that do not have tags, enter:

```
soamview task symping:901 -t "!" -s done
```

Note:

The "!" tag is a standalone reserved character and cannot be used to negate a regular expression.

-n counter

Returns the maximum number of records specified.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Display information for specific tasks that match a task tag

Display summary and brief task information for running tasks tagged as myTask.

```
soamview task sampleApp:101 -s running -t myTask
```

Display information for tasks that match a specified task state and submission time

Display summary and brief task information for currently running tasks that were submitted during the specified interval.

```
soamview task sampleApp:101 -s running -c 2009/01/21/10:00, 2009/01/21/11:00
```

Display information for tasks that match the specified task state during a specified time interval

Display summary and brief task information for tasks that were running during the specified interval. In this example, the system will return all tasks that were running between 10:00am January 21, 2009 and 2:00pm January 22, 2009 for session 101.

Note:

To locate the relevant session IDs, you should first display the sessions that were running during the specific interval and take note of the IDs .

```
soamview task sampleApp:101 -s running -c ,2009/01/22/14:00 -e 2009/01/21/10:00,-
```

task *application_name*: *session_ID*:*task_ID* [-l] [-u *user_name*] [-x *password*]

Displays brief and summary information for a specific task.

application

Specifies the name of the application. The application name is the same as it appears in the application profile. Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

session_ID

Specifies the Platform Symphony-assigned ID of the session as reported by `soamview session`.

task_ID

Specifies the Platform Symphony-assigned ID of the task as reported by `soamview task`.

-l

Provides detailed information.

-u user_name

Specifies the name of the user to connect to Platform Symphony for this command. If you are already logged on to Platform Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

Display information for a specific task

Display summary and brief information for task ID 4899.

```
soamview task sampleApp:101:4899
```

Time interval format

You use the time interval to define a start and end time for collecting the data to be retrieved and displayed. While you can specify both a start and an end time, you can also let one of the values default. You can specify either of the times as an absolute time, by specifying the date or time, or you can specify them relative to the current time.

Specify the time interval as follows:

start_time,end_time|start_time,-|-,end_time|start_time

Specify *start_time* or *end_time* in the following format:

[[year/][month/][day]/hour.minute]

Where:

- *year* is a four-digit number representing the calendar year.
- *month* is a number from 1 to 12, where 1 is January and 12 is December.
- *day* is a number from 1 to 31, representing the day of the month.
- *hour* is an integer from 0 to 23, representing the hour of the day on a 24-hour clock.
- *minute* is an integer from 0 to 59, representing the minute of the hour.

start_time,end_time

Specifies both the start and end times of the interval.

start_time,-

Specifies a start time, and lets the end time default to now.

,end_time

Specifies to start with the first logged occurrence, and end at the time specified.

start_time

Starts at the beginning of the most specific time period specified, and ends at the maximum value of the time period specified. For example, *2/* specifies the month of February—start February 1 at 00:00 a.m. and end at the last possible minute in February: February 28th at midnight.

Absolute time examples

Assume the current time is May 9 17:06 2008:

2008/5/8/13: = May 8 13:00 to May 8 13:59

2008/5/8/13:. = May 8 13:00 to the current time

2008/5/8/13:30 = May 8 13:00:00 to May 8 13:30:59

2008/5/8/13:30. = May 8 13:00:00 to the current time

1,8 = May 1 00:00 2008 to May 8 23:59 2008

,4 = the time of the first occurrence to May 4 23:59 2008

6 = May 6 00:00 2008 to May 6 23:59 2008

2/ = Feb 1 00:00 2008 to Feb 28 23:59 2008

/12: = May 9 12:00 2008 to May 9 12:59 2008

2/1 = Feb 1 00:00 2008 to Feb 1 23:59 2008

2/1. = Feb 1 00:00 to the current time

,2/10: = the time of the first occurrence to May 2 10:59 2008

2001/12/31,2008/5/1 = from Dec 31, 2001 00:00:00 to May 1st 2008 23:59:59

Relative time examples

`.-9,` = April 30 17:06 2008 to the current time

`,-2/` = the time of the first occurrence to Mar 7 17:06 2008

`.-9,-2` = nine days ago to two days ago (April 30, 2008 17:06 to May 7, 2008 17:06)

Here are some relative time examples expressed in a command string:

- Display sessions that were created 1 minute ago:

`soamview session sampleApp -c .-:1,`

- Display sessions that were created 2 hours ago:

`soamview session sampleApp -c .-:2,`

- Display sessions that were created 3 days ago:

`soamview session sampleApp -c .-3,`

symexec

Symexec supports the running of remote execution tasks using the Symphony infrastructure. An execution task is a child process executed by a Symphony service instance using a command line specified by a Symphony client.

Synopsis

symexec create [-a *application_name*] [-e *env1=value1[,env2=value2,...]*] [-p *preexecution_command*] [-q *postexecution_command*] [-u *user_name*] [-x *password*]

symexec send -s *session_ID* [-a *application_name*] [-e *env1=value1[,env2=value2,...]*] [-p *preexecution_command*] [-q *postexecution_command*] [-u *user_name*] [-x *password*] *command*

symexec fetch -s *session_ID* -n *counter* [-a *application_name*] [-t *timeout_in_seconds_for_fetch*] [-u *user_name*] [-x *password*]

symexec close -s *session_ID* [-a *application_name*] [-u *user_name*] [-x *password*]

symexec run [-a *application_name*] [-e *env1=value1[,env2=value2,...]*] [-p *preexecution_command*] [-q *postexecution_command*] [-t *timeout_in_seconds_for_fetch*] [-u *user_name*] [-x *password*] *command*

symexec -h | -V

Description

Use symexec as a Symphony client application to run executables, such as scripts, applications, or system calls. You cannot use symexec to run interactive graphical user interface programs. symexec requires an application and a service. The default, out-of-box application is symexec5. 1.

- Use **symexec create** to create an execution session in which to run execution tasks. The session stays open until it is explicitly closed.
- Use **symexec send** to send an execution task command in the execution session.
- Use **symexec fetch** to retrieve the statuses of finished execution tasks in the execution session. The statuses are retrieved as return codes or exception description and error codes.
- Use **symexec close** to close the execution session.
- Use **symexec run** to create an execution session, run a command, retrieve the results and close the session.

To run an execution command on Windows, specify the full name including the file extension.

Note:

It is recommended that you do not terminate the client (for example, using Ctrl + C). This could result in loss of tasks. If interrupts are required, specify a short timeout for fetch value.

Important:

If a symexec fetch command is in progress while a fetch or send command is issued for the same session from another command prompt window, the original fetch operation will abort and the session will detach from the original client

-h

Prints command usage to stdout and exits.

-V

Prints product version to `stdout` and exits.

Options

-a *application=name*

Specifies the name of the application to use. If you do not specify an application name, the application defaults to `symexec5.1`. If you specify an application name, it must match the application name in the corresponding application profile, and the application must be registered.

-e *env=value*

Sets the environment variables to be passed to the execution session. Specify as many environment variable/value pairs as required to define the environment.

To specify multiple environment variable/value pairs, separate the pairs with a comma.

-p *preexecution_command*

Specify a pre-execution command to run before running the executable command. If the pre-execution command is successful, the executable command is run. The command you specify must exist in the same location on every compute host on which it may run, or be accessible from the compute host. Specify an absolute path, or set the path in the `PATH` environment variable.

The pre-execution command context depends on command association, as follows:

Command	Pre-execution command context
create	session level
send	task level

-q *postexecution_command*

Specify a post-execution command to run. The execution service ensures that if the pre-execution command completes successfully, the post-execution command will always be executed, even if errors occur in the user's command during the session. The command you specify must exist in the same location on every compute host on which it may run, or be accessible from the compute host. Specify an absolute path, or set the path in the `PATH` environment variable.

The post-execution command context depends on command association, as follows:

Command	Post-execution command context
create	session level
send	task level

-u *user_name*

Specify the user ID under which to run the execution session.

-x *password*

Specify the password to authenticate the user ID.

-s session_ID

Specifies the unique ID of the session to operate upon.

-n counter

Specifies the number of tasks for which to fetch results.

If there are less than *n* task results available to be retrieved, `symexec fetch` waits until it can fetch all *n* tasks, or until the value set for `timeout_in_seconds_for_fetch` is reached.

If another `symexec send` or `symexec fetch` command is issued for the same session ID, the existing `symexec fetch` client is terminated, because only one client connection can be made to the session at a time.

Note:

Any task can only be fetched just once.

-t timeout_in_seconds_for_fetch

Specifies the number of seconds to wait for the session results.

If you do not specify a timeout, `symexec` waits indefinitely until all requested task outputs are received. If you specify 0 seconds, `symexec` returns immediately with or without the task output.

If the timeout expires and all of the results have still not been received, one of the following happens, depending on the subcommand the timeout is specified for:

- `symexec fetch`
Keep the session open and continue to run the task.
- `symexec run`

Close the session and terminate the task.

command

Specifies the execution command to run. The executable you specify must exist in the same location on every compute host on which it may run, or be accessible from the compute host. Specify an absolute path, or set the path in the PATH environment variable.

Restriction:

You cannot run a Windows internal command or a UNIX shell built-in command unless it is contained in a script.

Create an execution session to run a single command

```
symexec run -u user01 -x 12345 mycmd.exe
```

Creates an execution session using the default application `symexec5.1`, runs `mycmd.exe`, retrieves the results, and closes the session.

Create an execution session to run multiple commands

```
symexec create -u user01 -x 12345
symexec send -s 123 -u user01 -x 12345 mycmd.exe
symexec fetch -s 123 -n 1 -u user01 -x 12345
symexec send -s 123 -u user01 -x 12345 mycmd1.exe
```

- Creates an execution session using the default application symexec5. 1, and returns a session ID
- Sends mycmd.exe to the session
- Retrieve the results of the command
- Sends another command, mycmd1.exe to the session. The session remains open to receive other commands.

Retrieving stdout and stderr from execution tasks

The results from an execution task can be in the form of an exit code with optional stdout and stderr data files from the execution task's command. To retrieve the stdout and stderr data from a completed execution task, you must change its command to run under a shell so the command can redirect the stdout and stderr data to file(s). The file(s) can then be retrieved by either copying the file from the remote host (for example, with an FTP command) or by using a shared file system location.

The following examples demonstrate how to retrieve the stdout and stderr data from Windows and Linux hosts.

Windows:

In this example, the command will take the output of the "dir c:\\" command and place it in the dir_content.txt file. The command can be entered in the Start > Run textbox. The **cmd /c** opens a command shell and closes it after processing the string.

cmd /c dir c:\> c:\temp\dir_content.txt

Linux:

In this example, the command will take the output of the "ls /" command and place it in the root_content.txt file. The **sh -c** opens a command shell, which reads the commands from the string.

sh -c 'ls /> /tmp/root_content.txt'

symping

Sends workload to a cluster to test and verify that Symphony components are working and responsive.

Synopsis

symping

[-f *configuration_file*]
[-u *user_name*]
[-x *password*]
[-a *application_name*]
[-s *session_type*]
[-l [*summary_file*]]
[-T *taskoverhead*]
[-i *input_msg_size_in_bytes*]
[-o *output_msg_size_in_bytes*]
[-r *task_processing_time_in_milliseconds*]
[-c *common_data_size_in_bytes*]
[-m *number_of_tasks_per_testrun*]
[-n *number_of_testruns*]
[-t *client_type*]
[-g *session_tag*]
[-k *task_tag*]
[-d] [-z] [-p] [-j]
symping -h | -V

Description

Use the `symping` Symphony client to send test workload to Symphony to check that the cluster is working. You can also use this client to send different types of workload.

Without any options, `symping` tests for a complete workload cycle. In this type of test, workload is run through the cluster with full recoverability enabled (session and history). This type of run provides minimum, maximum, and average round-trip times. The average processing time averaged across all hosts is also displayed.

The default complete workload cycle configuration is the following:

- number of sessions—1
- number of tasks—20
- session type—RecoverableAllHistoricalData
- task processing time—50 milliseconds
- input message size—64 bytes
- output message size—128 bytes

The `symping` Symphony client is part of the Symphony diagnostic application. Application components are installed with the Symphony DE, and with Symphony. The diagnostic application is registered on all compute hosts in the `/SymTesting/Symping51` consumer. The diagnostic application `symping` is composed of the following:

- A service—`sympingservice`. The service sleeps for the task processing time. The service is always preloaded and holds one slot in the `ComputeHosts` group, even when idle. Use the `-r` and `-o` options to change the task processing time or the output message size.
- An application profile—`symping5.1.xml`.
- Location of `symping` executables and configuration:
 - Windows
 - 32-bit: `%SOAM_HOME%\5.1\win32-vc7\bin`
 - 64-bit: `%SOAM_HOME%\5.1\w2k3_x64-vc7-psdk\bin`
 - Linux

Located in the architecture-specific directory.

- 32-bit: For example, `SOAM_HOME/5.1/linux2.4-glibc2.2-x86/bin`
- 64-bit: For example, `SOAM_HOME/5.1/linux2.6-glibc2.3-x86_64/bin`

-h

Prints command usage to `stdout` and exits.

-V

Prints product version to `stdout` and exits.

symping options

-f configuration_file

Path and configuration file to use. The configuration file must be accessible from the machine on which the `symping` client is running.

Use this option to specify options in a configuration file instead of specifying them on the command line. If options are specified on the command line and a configuration file is specified, command-line options override those in the configuration file.

You can find a sample configuration file `symping.conf` in the `bin` directory in which the `symping` client is located.

-u user_name

Not applicable to Symphony DE.

Specifies the name of the user to connect to Symphony for this command. The user must have access to the consumer to which the application is registered.

If you are already logged on to Symphony using `soaml ogon`, for this command only the user name specified here overrides the user name entered in `soaml ogon`.

-x password

Not applicable to Symphony DE.

Specifies the user password to connect to Symphony for this command. If you are already logged on to Symphony using `soaml ogon`, for this command only the password specified here overrides the password entered in `soaml ogon`.

-a application_name

The name of the application that symping submits the workload to. The application name is the same as it appears in the application profile.

To run symping, the symping service binary must be configured in the application profile (it is configured and deployed out of box). If you have a mixed cluster, refer to your installation guide to ensure you have made the necessary modifications for symping to run.

Important:

The application name should have "symping" in its name. If you copy and rename this application, keep "symping" in the file name so that you can distinguish the applications related to the Symping client from those that are not.

Enclose the application name in double quotes (" ") if it contains spaces. Precede the application name with two dashes (--) if it starts with a dash.

-s session_type

Specifies the type of the session to create. The session type is coded in the symping client and is indicated in the application profile. Including history and recoverability increases the task and session overhead but more likely simulates a production environment.

Possible session types are:

- RecoverableAllHistoricalData: Session is recoverable and task and session history are set to "All". This is the default session type when you run symping without any options.
- RecoverableNoHistoricalData: Session is recoverable but no history (session or task) is kept.
- UnrecoverableAllHistoricalData: Session is unrecoverable and no history (session or task) is kept.
- UnrecoverableNoHistoricalData: Session is unrecoverable and no host history (task or session) is kept. This is the default session type if you choose to run symping configured for task overhead with the `-T taskoverhead` option.

-l [summary_file]

If you specify `-l` without a summary file name, creates a summary file for symping output in the directory in which symping is run with the name `symping.summary.date_stamp.hhmmss`.

If you specify the path and name to assign to the summary file, generates output in the specified summary file. If the summary file exists, appends output to the existing summary file.

-T taskoverhead

Optimizes the symping configuration to calculate task overhead—the Symphony overhead for processing a single task.

The default configuration is the following:

- number of sessions—1
- number of tasks—20
- session type—UnrecoverableNoHistoricalData
- task processing time—0 milliseconds
- input message size—1024 bytes
- output message size—1024 bytes
- common data—0 bytes

Results include overhead for longest and shortest task, average overhead, and average processing time.

-i input_msg_size_in_bytes

Specifies the size of input messages for all tasks in the session, in bytes.

If you do not specify this option, the input message size for tasks is 64 bytes.

-o output_msg_size_in_bytes

Specifies the size of all output messages for all tasks in the session, in bytes.

If you do not specify this option, the output message size for tasks is 128 bytes.

-r task_processing_time_in_milliseconds

Specifies how long each task in the session runs, in milliseconds.

Task processing time is a time duration. Task processing time is the pure task processing time, without overhead. It is the time spent on the `onInvoke()` call. The clock is started when SIM calls `onInvoke()` in the service code. The clock ends when the `onInvoke()` returns. If you do not specify the task processing time, tasks run for 50 milliseconds.

-c common_data_size_in_bytes

Specifies the size of the common data that is associated with the session and passed from the client to the service, in bytes.

If you do not specify the size of common data, there is no common data passed from the client application to the service.

-m number_of_tasks_per_testrun

Specifies the number of tasks that are sent per test run.

If you do not specify the number of tasks, the client application creates one session with 20 tasks.

-n number_of_testruns

Specifies the number of test runs that are created. A test run indicates sending input and retrieving output.

For example, in one test run with 10 tasks, 1 session is created and 10 tasks sent.

In 2 test runs with 10 tasks, 1 session is created, 10 tasks are sent and output retrieved, then 10 more tasks are sent and output retrieved.

If you do not specify the number of test runs, the client application creates 1 test run.

-t client_type

Specifies whether tasks are sent and output retrieved synchronously or asynchronously. Valid values are `sync` and `async`.

If you do not specify the client type, tasks are sent and output retrieved asynchronously.

-g session_tag

Adds the specified session tag to any session created by symping. A session tag that is shared among sessions provides the ability to query or control these related sessions with a single action.

-k task_tag

Adds the specified task tag to all tasks submitted with symping. The task tag can later be used to filter only those tasks that share the given tag.

-d

Does not display detailed output for each task.

If you do not use this option, detailed output for each task is displayed.

-z

Specifies to consume actual CPU cycles on compute hosts.

If you do not specify this option, the service sleeps for the specified task processing time, not consuming actual CPUs on the compute host.

-p

Symphony DE only.

Displays compute host status in Symphony DE.

-j

Generates a service log file on compute hosts on which the symping service runs. You can retrieve and view the log file with the Platform Management Console.

The log is written to:

- Windows— %SOAM_HOME%\work*application_name**session_ID*
- Linux— \$SOAM_HOME/work/*application_name*/*session_ID*

The log file is named according to session and task IDs: *sessionID.taskID*.

Without this option, no service log is generated.

Output

The following are definitions for terms used in the symping output:

- Task overhead:

Duration. Symphony overhead for processing a single task. It is calculated as follows: Task roundtrip minus task processing

- Task processing:

Duration. Pure task processing time (no overhead): time spent on the service onInvoke() call. The clock is started when SIM calls onInvoke() in the service code. The clock ends when onInvoke() returns.

- Task roundtrip:

Duration. Time for a full task cycle (overhead + processing) of a single task. It is the time from the send task input call to the return of the task output.

Task round trip is defined as: Task output received minus task input sent

- Total task roundtrip:

Duration. Time for a full task cycle (overhead + processing) for all tasks in a session. It is the time from the send task input call to the return of the task output.

Total task roundtrip is defined as: Last task output received minus first task input sent

- Test roundtrip:

Duration. Clock is started when the user runs the symping command. Clock ends when symping returns with the client application's results and returns control to user. This time duration includes initialization, connection time, all phases before session creation time.

Examples

Test for a complete workload cycle

```
symping
```

Test for task overhead

Use the -T taskoverhead option to optimize configuration to calculate system overhead.

```
symping -T taskoverhead
```

Create a summary file with symping output

Use the -l option to generate a summary file with symping output.

```
symping -l
```

Generate a service log file on compute hosts

Use the -j option to generate a service log file on compute hosts.

```
symping -j
```


Run one session with 10 tasks

```
symping -m 10
```

Run 2 sessions, 1000 tasks each, 1 millisecond in length

To run more than one session, you need to run `symping` twice

```
symping -m 1000 -r 1
```

```
symping -m 1000 -r 1
```

Run 30 tasks, keep the session open, then run 30 more tasks

```
symping -m 30 -n 2
```

Use common data of 150 bytes, input messages of 20 bytes, output messages of 20 bytes

```
symping -c 150 -i 20 -o 20
```

Run one session with 10 tasks, 1 millisecond in length, with common data of 150 bytes, and input and output messages of 20 bytes

```
symping -m 10 -r 1 -c 150 -i 20 -o 20
```


symping

P A R T



Application Profile

Application profile reference

About application profiles

The application profile is used to change application parameters without requiring code changes. The application profile defines application behavior within Symphony. It provides the information Symphony needs to run services and manage workload. Each application name in the profile must be unique in the cluster and associated with one consumer. At any time, there can be only one enabled application per consumer.

You can find example application profiles in the samples directories. You can also refer to the application profile templates to create your own application profiles. You can find templates in the `$$SOAM_HOME/5.1/samples/Templates` directory.

Application profile fragment

With dynamic application updates, you can add a new service by registering a *profile fragment* (part of an application profile) that contains either new session types in the SessionTypes section, new Service sections, or both. When you register a profile fragment that adds a session type or Service section, running workload continues to run. The session director updates the session manager with the new version of the application profile.

Note:

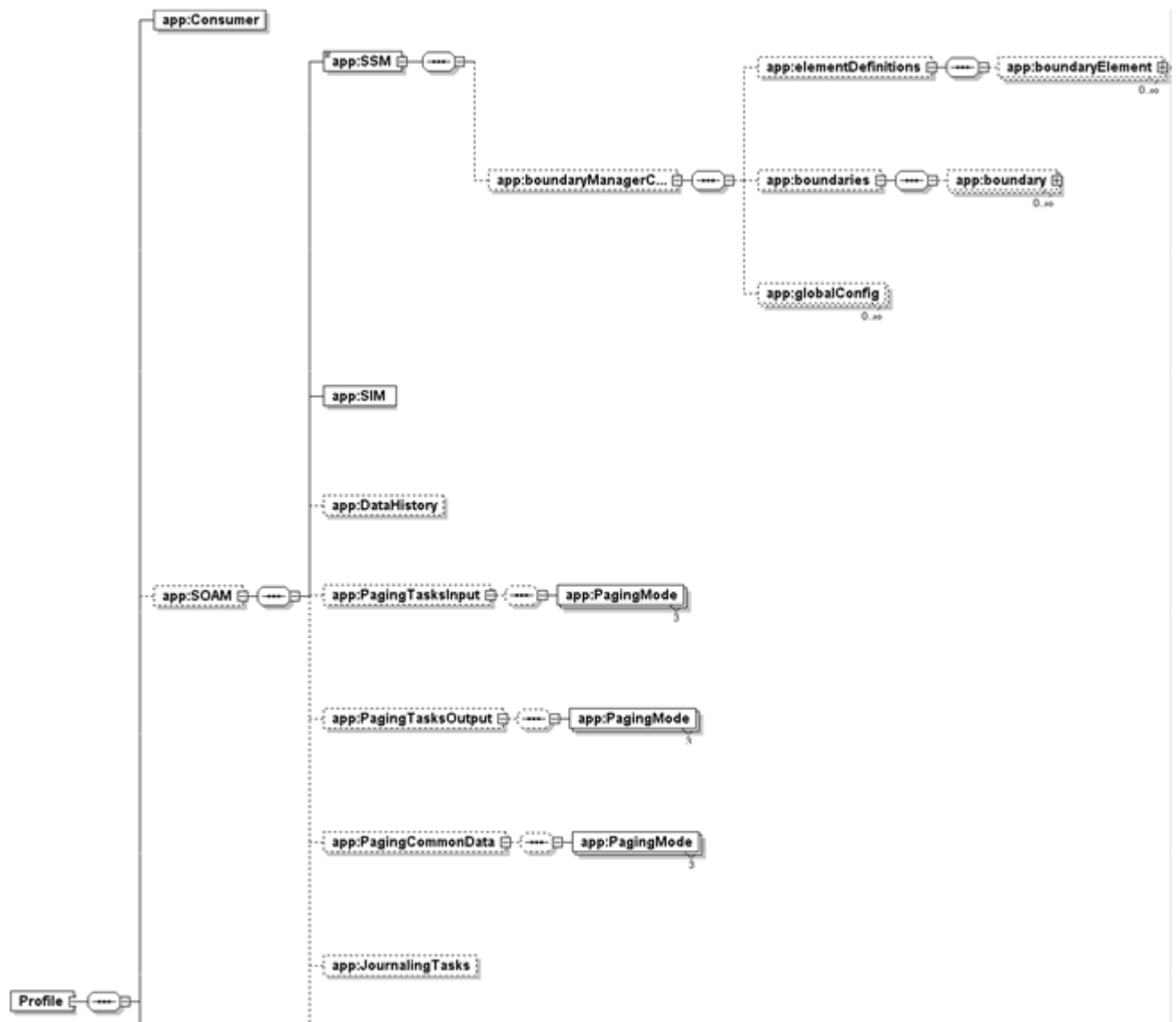
You can add, update, or remove session types or Service sections or both by modifying an existing application in PMC using **Dynamic Configuration Update**. In this case, you are not required to create fragments manually. See the PMC Help or Platform Application Development Guide, available from the Platform Knowledge Center, for more details.

You can refer to the profile fragment templates to create your own profiles. You can find templates in the `$$SOAM_HOME/5.1/samples/Templates` directory. However, this is not a recommended way for updating application parameters.

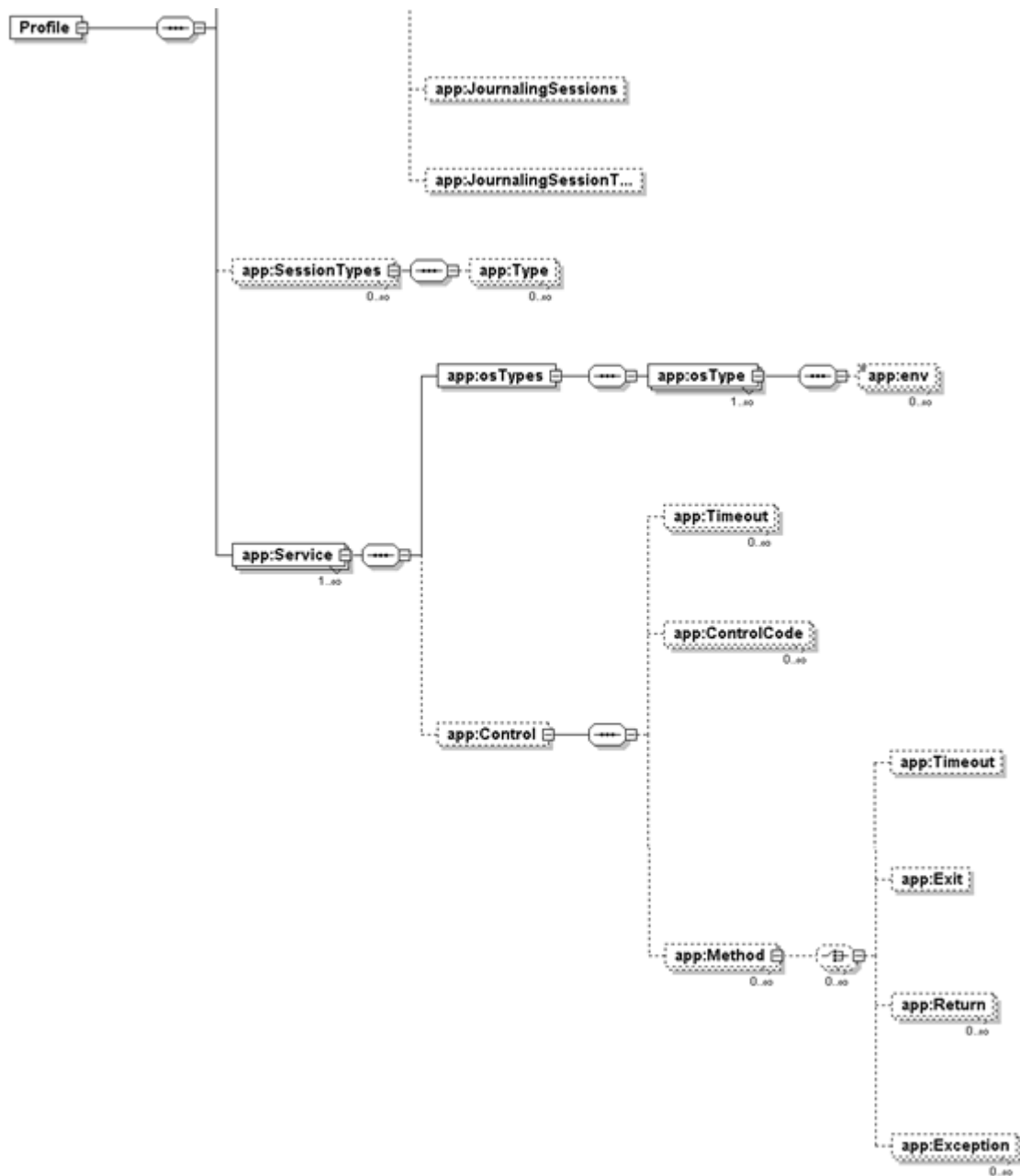
When you update a Service section or session type, the session manager only terminates workload that uses the updated service(s) and/or session type(s).

Application Profile Structure

Application profile structure (part 1)



Application profile structure (part 2)



Profile

name

For internal system use only. Not used by the system.

version

Specifies the version of the application profile.

Application profile reference

Where used

Profile

Required/Optional

Optional

Valid values

3.1 or higher

Default value

3.1

Example

version="5.1"

Related Attributes

Version (SOAM section)—Symphony middleware version.

Consumer section

Settings related to the application and resources assigned to the consumer associated with the application.

applicationName

Name of the application associated with the consumer. The application name must be unique within the cluster.

Enclose the application name in double quotes if it contains spaces, for example, name="Commodity Derivatives App".

Where used

Consumer

Required/Optional

Required

Valid values

In the Platform Management Console: Accepts "a-z, A-Z, 0-9 or spaces. Do not use "al l " as an application name—it is a reserved word.

On the command line: Accepts all ASCII printable characters except "\" / : * ? ' | & < or > ". Does not accept leading or trailing spaces in application names. Do not use "al l " as an application name—it is a reserved word.

consumerId

Name and path of the consumer with which to associate this application. The consumer must already exist in Symphony. The consumer you associate with the application determines how Symphony allocates resources to the application.

The consumer ID includes the path to the consumer within the consumer tree.

The consumer must be a leaf consumer. A leaf consumer always resides at the lowest level of the consumer tree.

A unique consumer ID must be specified in Symphony, but it is not required to be pre-defined in SymphonyDE. SymphonyDE uses the consumer ID to associate applications and service packages. Only one application can be enabled within a consumer ID and service packages must be deployed to the same consumer ID.

Where used

Consumer

Required/Optional

Required

Valid values

Alphanumeric string that does not include any special characters.

The consumer ID begins with a forward slash (/) and each level is separated from the next with a forward slash.

Example

consumerId="/New York Cluster/Capital Markets/Derivatives"

resourceGroupName

Resource group from which resources are requested to run service instance managers and services to perform computations for clients. The resource group should contain compute hosts, not management hosts. The resource group name can contain multiple resource groups, e.g., "rg1 rg2 rg3".

Not required for Symphony DE.

Where used

Consumer

Required/Optional

Optional

Default value

" " (empty)—all resource groups which are associated with the consumer will be used.

Related attributes

- numOfSlotsForPreloadedServices—Defines the number of slots required for service instances that are prestarted if preStartApplication is true. The number cannot be greater than the number of slots available in the resource groups, defined by resourceGroupName
- resReq—Defines the resources an application requires
- resourceGroupFilter—Defines a list of resource groups from which the session can use resources. This list either matches or is subset of the resource groups specified in resourceGroupName.

resReq

String that describes criteria for selecting resources allocated to a consumer for compute hosts.

Ignored in Symphony DE.

Where used

Consumer

Required/Optional

Optional

Default value

select (!mg)

Related attributes

- resourceGroupName—Defines the resources available to an application

Synopsis

"select(select_string)" "select(select_string) order(order_string)"

Description

A resource requirement string describes the criteria for defining a set of resources for compute hosts.

Note:

If you specify a resource requirement, ensure you indicate "select (!mg)" in addition to your resource requirement string. This is so that the system only selects compute hosts to run work, not management hosts.

The entire resource requirement string cannot contain more than 512 characters.

Options

select(select_string)

Specifies the criteria for selecting the resources. The selection string filters out the resources that do not meet the criteria, resulting in a list of one or more eligible resources.

The parentheses must be typed as shown.

order(order_string)

Specifies the sort order for selecting the best resource from the list of eligible resources. The most appropriate resource is listed first, the least is listed last.

The parentheses must be typed as shown.

select

select synopsis

select(expression)select(expression operator expression)select((expression operator expression) operator expression)

select description

The selection string is a logical expression used to select one or more resources to match one or more criteria. Any resource that satisfies the criteria is selected.

resource_name operator value

resource_name

The following resources can be used as selection criteria.

Static resources

Static resources are built-in resources that represent host information that does not change over time, such as the maximum RAM available to user processes or the number of processors in a machine. Most static resources are determined at start-up time, or when hardware configuration changes are detected.

Static resources can be used to select appropriate hosts based on binary architecture, relative CPU speed, and system configuration.

Note:

The resources ncpus, ncores, nprocs, nthreads, maxmem, maxswp, and maxtmp are not static on UNIX hosts that support dynamic hardware reconfiguration.

Index	Measures	Units	Determined by
type	host type	string	configuration
model	host model	string	configuration
hname	host name	string	configuration
cpuf	CPU factor	relative	configuration
server	host can run remote jobs	Boolean	configuration
rexpri	execution priority	nice(2) argument	configuration
ncpus	number of processors	processors	LIM
ndiks	number of local disks	disks	LIM
maxmem	maximum RAM	MB	LIM
maxswp	maximum swap space	MB	LIM
maxtmp	maximum space in /tmp	MB	LIM

CPU factor (cpuf)

The CPU factor is the speed of the host's CPU relative to other hosts in the cluster. If one processor is twice the speed of another, its CPU factor should be twice as large. CPU factors are defined by the cluster administrator. For multiprocessor hosts, the CPU factor is the speed of a single processor.

Server

The server static resource is Boolean. It has the following values:

- 1 if the host is configured to run jobs from other hosts.
- 0 if the host is a client for submitting jobs to other hosts.

operator

The following operators can be used in selection strings. If you are using the selection string in an XML format, you must use the applicable escape characters in the XML Equivalence column. The operators are listed in order of decreasing precedence:

Operator	XML Equivalent	Syntax	Meaning
!	n/a	!a	Logical NOT: 1 if a==0, 0 otherwise
*	n/a	a*b	Multiply a and b
/	n/a	a / b	Divide a by b

Operator	XML Equivalent	Syntax	Meaning
+	n/a	a+b	Add a and b
-	n/a	a-b	Subtract b from a
>	>	a > b	1 if a is greater than b, 0 otherwise
<	<	a < b	1 if a is less than b, 0 otherwise
>=	>=	a >= b	1 if a is greater than or equal to b, 0 otherwise
<=	<=	a <= b	1 if a is less than or equal to b, 0 otherwise
==	n/a	a == b	1 if a is equal to b, 0 otherwise
!=	n/a	a != b	1 if a is not equal to b, 0 otherwise
&&	&&	a && b	Logical AND: 1 if both a and b are non-zero, 0 otherwise
	n/a	a b	Logical OR: 1 if either a or b is non-zero, 0 otherwise

value

Specifies the value to be used as criteria for selecting a resource. Value can be numerical, such as when referring to available memory or swap space, or it can be textual, such as when referring to a specific type of host.

Simple selection expression using static resources

The following example selects resources with total memory greater than 500MB:

```
select (maxmem > 500)
```

Compound selection expression using static resources

The following example selects resources with total memory greater than 500MB and total swap space greater than or equal to 300MB:

```
select (maxmem > 500 & & maxswp >= 300)
```

order

Sorts the selected resources into an order of preference according to the values of the resources.

order synopsis

order(*order_string*)

order description

The order string acts on the results of a select string, sorting the selected resources to identify the most favorable resources, and eliminate the least desirable resources.

Resources are sorted into ascending order based on the result of an arithmetical expression.

The result orders the resources from smallest to largest.

order_string

Specify an arithmetic expression that expresses the desired outcome.

You can create an expression using operators and numbers.

Use the following operators:

+	add
-	subtract-
*	multiply
/	divide

Order resources based on available swap space

The following example orders the selected resources based on their available swap space, but sorts by descending order, ordering hosts with the greatest amount of available swap space first:

```
order (0- swp)
```

policy

Specifies the workload scheduling policy used. The scheduling policy determines how and when Symphony allocates resources to run services for the application.

Where used

Consumer

Required/Optional

Optional

Default value

R_Proportion

Valid values

- R_Proportion

Summary

A new session can get CPU slots immediately without waiting for current sessions to finish. Sessions simultaneously share a portion of the total

CPU slots assigned to the application. CPU slots are proportionally assigned to sessions based on priority.

Disadvantage	This scheduling policy can have performance issues if it takes a long time to bind a service to a session since the service instances may toggle among open sessions.
Initial slot allocation	<p>Symphony assigns CPU slots first to higher priority sessions to ensure they finish faster.</p> <p>CPU slots are assigned proportionally to sessions according to their priority.</p> <p>CPU factors are taken into account during allocation. The most powerful idle CPU is allocated first to higher priority sessions.</p> <p>The CPU factor of the compute host is defined in the <code>conf\ven_resource.conf</code> file under the Symphony DE installation directory, and is detected by Platform EGO in Symphony cluster when Symphony starts on the host.</p>
Slot reallocation based on workload	<p>Yes.</p> <p>When a session does not need all slots allocated to it, the slots are released and reallocated to other sessions.</p> <p>Slots are only reallocated to sessions with pending tasks.</p>
Higher priority sessions can take slots away	<p>Yes.</p> <p>As soon as a higher priority session needs more resources, Symphony can allocate slots that are being used by lower priority sessions to the higher priority session after the tasks running on the slots are finished.</p>
Service instance unbound from session	Service instances are unbound from the session based on workload, when there are not enough pending tasks for all the service instances currently allocated.

- **R_MinimumServices**

Summary	<p>Use this scheduling policy when you are using common data so that service instances will be reused for tasks in the same session, eliminating the need to reload data for each task.</p> <p>With this scheduling policy, a minimum number of CPU slots is allocated to a session regardless of workload or priority.</p>
Initial slot allocation	<p>Symphony attempts to allocate slots to the session to satisfy the minimum configured number of service instances.</p> <p>If there are enough slots to meet the minimum number of instances for all open sessions:</p> <ul style="list-style-type: none"> • Minimum instances are allocated to sessions based on session submission time. • Leftover slots are allocated to sessions proportionally based on session priority and submission time. <p>If there are not enough slots to meet the minimum number for all sessions:</p> <ul style="list-style-type: none"> • Higher priority sessions get all minimum instances satisfied first. <p>CPU factors are taken into account. CPU slots are evenly distributed among sessions based on speed so that all sessions get a proportion of the fastest CPUs.</p>

Slot reallocation based on workload	<p>Partial.</p> <p>A minimum number of CPU slots is always allocated to a session, regardless of workload.</p> <p>Additional CPU slots are distributed proportionally to other open sessions with pending tasks according to priority and submission time.</p> <p>When there is no workload in the session, CPU slots not serving the minimum required service instances are distributed proportionally to other open sessions with pending tasks according to priority and submission time.</p> <p>Sessions with no pending tasks do not receive any slots, except the required minimum service instances.</p>
Higher priority sessions can take slots away	<p>Partial.</p> <p>A session always retains the required CPU slots to serve the minimum number of service instances, regardless of session priority. CPU slots serving the minimum number of service instances are not released.</p> <p>CPU slots not serving the minimum number of service instances are released and distributed to other sessions proportionally based on priority.</p>
Service instance unbound from session	<p>The minimum number of service instances are always bound to the session until the session is suspended, killed, or closed.</p> <p>Service instances additional to the minimum number are unbound from the session as needed, based on the priority and pending workload of all sessions.</p>

• R_PriorityScheduling

Summary	<p>A new session with the highest priority can get as many CPU slots as it needs. CPU slots are only allocated to other sessions when the highest priority session does not have any pending tasks. If preemption is enabled, sessions with the highest priority can preempt lower priority sessions. Following preemption, the resource is assigned to the highest priority session, and the task that was preempted is rerun.</p>
Initial slot allocation	<p>Symphony assigns all available CPU slots first to the highest priority session to meet the demand and ensure the session finishes as soon as possible.</p> <p>If there are more slots available than the highest priority session needs, the session with the next highest priority gets the surplus slots. This pattern of overflow slot allocation continues, in turn, to other sessions in a descending priority order.</p> <p>If two sessions have equal priority, the session that was created first is considered to have the higher priority.</p>
Slot reallocation based on workload	<p>Yes.</p> <p>When a session no longer needs all slots allocated to it, the slots are reallocated to a session with the next highest priority.</p> <p>Slots are only reallocated to sessions with pending tasks.</p>

Higher priority sessions can take slots away	<p>Yes.</p> <p>As soon as a higher priority session needs more resources, Symphony can allocate slots that are being used by lower priority sessions to the higher priority session after the currently running task is finished. If preemption is enabled for the higher priority session, the current running task of the lower priority session is preempted.</p>
Service instance unbound from session	Service instances are unbound from the session when there are no pending tasks in the session.

Related attributes

SessionType:

- **priority**—Determines how tasks are assigned resources.
- **preemptive**—Determines whether higher priority sessions preempt lower priority sessions.
- **minServices**—Determines the minimum number of CPU slots required for sessions.

Consumer:

- **taskHighWaterMark**—Defines the maximum ratio of unprocessed tasks to CPU slots before more resources are requested.
- **taskLowWaterMark**—Defines the minimum ratio of unprocessed tasks to CPU slots before unused resources are released.
- **resourceBalanceInterval**—Defines when Symphony next checks to determine whether resources need to be requested or released from Platform EGO.
- **sessionSchedulingInterval**—Determines when resources are balanced among sessions.

taskHighWaterMark

Applies to the whole application. The ratio represents the total number of pending and running (unprocessed) tasks in open sessions to service instances. The value of **taskHighWaterMark** is fixed at 1; this means that for every unprocessed task, 1 CPU slot is requested (assuming the session has a service-to-slot ratio of 1:1).

In Symphony DE, the number of CPU slots on a host is configured in the `conf/vem_resource.conf` file under the Symphony DE installation directory.

In Symphony, the number of CPU slots on a host is configured in the `ResourceGroups.xml` file through the PMC.

Where used

Consumer

Default value

1.0 (not adjustable)

Related attributes

- **resourceBalanceInterval**—Defines when Symphony next checks to determine whether resources need to be requested or released from Platform EGO
- **taskLowWaterMark**—Defines the minimum ratio of unprocessed tasks to CPU slots before unused resources are released

- `serviceToSlotRatio`—Defines the number of slots required to run a service instance

taskLowWaterMark

Applies to the whole application. Ratio is used to determine whether idle CPU slots are released and made available for other applications to use. Once the ratio of unprocessed tasks to service instances falls below the `taskLowWaterMark`, resources are released and made available for other applications to use.

If `taskLowWaterMark` is 1.0, any resources the application releases must be idle. For example, you have thousands of unprocessed tasks. 100 CPU slots are allocated to the Session Manager. Note that this example assumes the session has a service-to-slot ratio of 1:1.

- When the number of unprocessed tasks reaches 200, the ratio of unprocessed tasks to CPU slots is $200/100 = 2$. The Session Manager does not release any CPU slots.
- When the number of unprocessed tasks reaches 100, the ratio of unprocessed tasks to CPU slots is $100/100 = 1$. The Session Manager does not release any CPU slots.
- When the number of unprocessed tasks drops below 100, the Session Manager releases slots until the ratio becomes greater than 1.

If `taskLowWaterMark` is equal to 0, the application never returns resources voluntarily.

Where used

Consumer

Required/Optional

Optional

Valid values

0 or 1

Default value

0

Related attributes

- `resourceBalanceInterval`—Defines when Symphony next checks to determine whether resources need to be requested or released from Platform EGO
- `taskHighWaterMark`—Defines the maximum ratio of unprocessed tasks to CPU slots before more resources are requested
- `serviceToSlotRatio`—Defines the number of slots required to run a service instance

resourceBalanceInterval

Minimum number of seconds between checks to determine whether the session manager should release unused resources or request additional resources for the application.

Where used

Consumer

Valid values

1 or more (if 0 is defined, the value of 1 is substituted)

Required/Optional

Optional

Default value

5 seconds

Related attributes

- `resReq`—Defines the resources an application requires
- `resourceGroupName`—Defines the resource groups available to an application
- `sessionSchedulingInterval`—Determines when resources are balanced among sessions
- `taskHighWaterMark`—Defines the maximum ratio of unprocessed tasks to CPU slots before more resources are requested
- `taskLowWaterMark`—Defines the minimum ratio of unprocessed tasks to CPU slots before unused resources are released

sessionSchedulingInterval

Minimum number of milliseconds between checks to determine whether compute resources need to be reallocated among sessions.

The system reallocates resources according to application session priority, scheduling policy, and pending tasks.

Where used

Consumer

Required/Optional

Optional

Valid values

1 or more

Default value

500 milliseconds

Related attributes

- `policy`—Defines how resources are shared.
- `priority`—Determines how tasks are assigned resources.
- `taskLowWaterMark`—Defines the minimum ratio of unprocessed tasks to CPU slots before unused resources are released
- `taskHighWaterMark`—Defines the maximum ratio of unprocessed tasks to CPU slots before more resources are requested

preStartApplication

Specifies whether the session manager, service instance manager, and service instances are prestarted for enabled applications when Symphony is started, or when an application is enabled.

Prestarting the session manager provides quicker response to the first client request for this application. Prestarting the service instances manager and service instances provides quicker response to tasks.

Where used

Consumer

Required/Optional

Optional

Valid values

true | false

Default value

false

Related attributes

- `numOfSlotsForPreloadedServices`—Defines the number of service instance slots required for a prestarted application. The number of slots cannot be greater than the number of slots available in the resource group, defined by `resourceGroupName`

numOfPreloadedServices (deprecated)

Used with pre-start application. Defines the number of service instances that are prestarted if `preStartApplication` is true.

One service instance manager is started for every service instance. One service instance can run on each CPU slot.

Note:

For Symphony version 5.0 and higher, if the value is set to more than 10000 , it will be reduced to 10000 and the SSM will give a WARN message.

Where used

Consumer

Required/Optional

Optional

Valid values

1 or greater

Default value

1

Related attributes

- `preStartApplication`—Services are preloaded only if `preStartApplication` is true
- `resourceGroupName`—Defines the resources available to an application

numOfSlotsForPreloadedServices

Used with pre-start application. Defines the number of slots required for service instances that are prestarted if `preStartApplication` is true.

Where used

Consumer

Required/Optional

Optional

Valid values

1 or greater up to a maximum of 10000.

Default value

1

Related attributes

- `preStartApplication`—Services are preloaded only if `preStartApplication` is true
- `resourceGroupName`—Defines the resources available to an application

enableStandbyService

Specifies whether resources that have been allocated to the application will have standby services running when slots are released by the SSM. Note that standby services are not supported by Symphony DE. In this case, standby service configuration is ignored by Symphony and a warning message is logged.

Where used

Consumer

Required/Optional

Optional

Valid values

true | false

Default value

false

transientDisconnectionTimeout

Number of seconds the session manager waits for the client to reconnect when the connection between the client and session manager is broken.

When `abortSessionIfClientDisconnect` is true, sessions which are open on the client connection are aborted if a disconnected client does not reconnect to the session manager.

The `transientDisconnectionTimeout` counter is reset when a client connects or re-connects to the session manager. Reconnection is done by the Symphony Client API, which is transparent to the client application.

Where used

Consumer

Required/Optional

Optional

Valid values

1 or more

Default value

30 seconds

Related attributes

- `abortSessionIfClientDisconnect`—Specifies whether the session is aborted if the session manager detects that the connection between the client and the session manager is broken.

flushDataAsap

Used for recoverable sessions. Specifies whether or not the session manager caches data before writing to disk.

When set to true, data is not cached, it is immediately written to disk. When set to false, data is cached before it is written to disk. Note that the setting in `flushDataAsap` does not affect any operating system configuration. You need to disable write-behind caching for your operating system. On Windows, see <http://support.microsoft.com/kb/247485> for more details. On Linux, contact your system administrator.

Important:

Setting this parameter to true may substantially degrade performance because system cannot continue with its next operation until the data is written to the disk.

Where used

Consumer

Required/Optional

Optional

Valid values

true | false

Default value

false

ioRetryDelay

Specifies the amount of time to wait, in seconds, before retrying an I/O operation after a previous failure.

Where used

Consumer

Required/Optional

Optional

Valid values

Non-negative integer

Default value

1 second

ssmRecoveryTime

For internal system use only.

delaySlotRelease

Specifies the amount of time, in seconds, that unassigned services will remain idle with the application before releasing their corresponding slot to EGO (and terminating the service). An unassigned service is defined as a service not assigned to any session. If new workload arrives before the configured time elapses, these unassigned services can be used to handle the new workload. Once an unassigned service becomes assigned to a session again, its idle timer is reset.

If delaySlotRelease is configured at the service-level, it overrides the application level setting for that service.

Where used

Consumer

Required/Optional

Optional

Valid values

Non-negative integer

Default

0 seconds

Related attributes

- delaySlotRelease—Specifies the delaySlotRelease time at the service level

enableSelectiveReclaim

Specifies whether selective reclaim is enabled.

A value of "true" means selective reclaim is enabled. Selective reclaim allows you more control over which task is interrupted during resource reclaim, because the system uses preemption rank and preemption criteria (both configurable) to determine which host supplies the resource.

Without selective reclaim, the host is selected at random but the SSM uses preemption criteria and preemption rank to choose which slots are reclaimed on that host.

Where used

Consumer

Required/Optional

Optional.

Valid values

true | false

Default

false

preemptionCriteria

Specifies the policy used to decide which task to interrupt to reclaim a resource.

This setting affects the resource reclaim process when the system has identified multiple sessions or tasks that could provide a resource, even after considering preemption rank or session priority.

If the value is set to MostRecentTask, the system selects the task with the least run time, and reclaims the resource used by that task. If multiple tasks have the same run time, the system selects any one at random. If multiple tasks run on a slot, consider the cumulative run time of all tasks using the slot.

If the workload scheduling policy (session scheduling policy) is proportional or minimum services, and the value is set to Default, the system selects the most overallocated session, and reclaims a resource used by that session. If multiple sessions are equally over-allocated, the system selects any one at random. If no session is over-allocated, select the least under-allocated instead. Task selection is random.

If the workload scheduling policy (session scheduling policy) is priority, and the value is set to Default, the system selects a task at random.

There is less system overhead if you choose Default.

Where used

Consumer

Required/Optional

Optional.

Valid values

Default | MostRecentTask

Default

Default

schedulingAffinity

Determines whether resource-aware scheduling for sessions and/or data-aware scheduling for tasks are enabled. When set to ResourceAware, the SSM collects host attributes and evaluates them against a resource preference expression for the session. When set to DataAware, the SSM collects data attributes of services instances and hosts and evaluates them against a user-defined preference expression for each task. Setting the attribute to DataAware automatically enables resource-aware scheduling for sessions.

Where used

Consumer

Required/Optional

Optional

Valid values

ResourceAware | DataAware | None

Default

None

preemptionScope

Determines whether sessions in the application can preempt sessions of lower or equal rank, or only preempt sessions of lower rank when preemptive is set to true.

Where used

Consumer

Required/Optional

Optional.

Valid values

LowerRankedSessions | LowerOrEqualRankedSessions

Default

LowerOrEqualRankedSessions

Related attributes

- preemptive—Specifies whether a session can preempt other sessions when it has workload to run.

slotsThresholdForStandbyServices

Maximum number of slots that standby services can run on for the application. If the number of slots exceeds this threshold, Symphony will release the excess slots at the next scheduling interval.

Consumer section

Where used

Consumer

Required/Optional

Optional.

Valid values

Integer, -1 (infinite) to 10000

Default

-1

Related attributes

- `enableStandbyService`—Specifies whether resources that have been allocated to the application will have standby services running when slots are released by the SSM.

SOAM section

The SOAM section contains configuration related to system behavior such as session manager, service instance manager, and how Symphony writes data to disk.

version

The version of the Symphony middleware.

Where used

SOAM

Required/Optional

Optional—When the version is not included, the application will run on the latest version of Symphony available in the cluster.

Valid values

3.1 or higher

Example

```
version="5.1"
```

SOAM>SSM section

Configuration related to session manager behavior.

startUpTimeout

Number of seconds to wait for the session manager to start up before the session director considers it as timed out.

When the process times out, the session director requests EGO to replace the session manager host and tries again to start a new session manager.

The session manager is started when Symphony is started if prestartApplication is true for the enabled application. Otherwise, the session manager is started when a client connects to an enabled application, which does not have running session manager and prestartApplication is false.

Where used

SOAM > SSM

Required/Optional

Optional

Valid values

1 or more

Default value

60 seconds

Related attributes

preStartApplication—Influences when the session manager and service instance manager are started, when Symphony starts or when Symphony receives a request from a client of this application

shutdownTimeout

Number of seconds to wait for the session manager to shut down before the session director considers it as timed out and restarts the session manager.

The session manager should shut down within the timeout period when the related application is unregistered or disabled.

Where used

SOAM > SSM

Required/Optional

Optional

Valid values

1 or more

Default value

300 seconds

disableFlowControl

For internal system use only.

resReq

String that describes criteria for selecting resources allocated to the session director to start the session manager.

This is not applicable for Symphony DE.

Where used

SOAM > SSM

Required/Optional

Optional

Default value

" " (empty)—SSM can run on any hosts in the resource groups configured in `resourceGroupName` in the SOAM > SSM section. By default, the `resourceGroupName` is `ManagementHosts`.

Synopsis

"select(select_string)" "select(select_string) order(order_string)"

Related Attributes

- resourceGroupName in SOAM > SSM—Resource group from which resources are requested to run session managers.

Description

A resource requirement string describes the criteria for defining a set of resources to run session managers. Session managers should run on management hosts. If you have changed the default value of resourceGroupName, you must indicate the select string

```
"select (mg) "
```

so that only management hosts are selected to run the session managers. Running the session manager on a management host is essential for reliability and recovery.

The entire resource requirement string cannot contain more than 512 characters.

Options

select(select_string)

Specifies the criteria for selecting the resources. The selection string filters out the resources that do not meet the criteria, resulting in a list of one or more eligible resources.

The parentheses must be typed as shown.

order(order_string)

Specifies the sort order for selecting the best resource from the list of eligible resources. The most appropriate resource is listed first, the least is listed last.

The parentheses must be typed as shown.

select

select synopsis

```
select(expression)select(expression operator expression)select((expression operator expression) operator expression)
```

select description

The selection string is a logical expression used to select one or more resources to match one or more criteria. Any resource that satisfies the criteria is selected.

```
resource_name operator value
```

resource_name

The following resources can be used as selection criteria.

Static resources

Static resources are built-in resources that represent host information that does not change over time, such as the maximum RAM available to user processes or the number of processors in a machine. Most static resources are determined at start-up time, or when hardware configuration changes are detected.

Static resources can be used to select appropriate hosts based on binary architecture, relative CPU speed, and system configuration.

Note:

The resources ncpus, ncores, nprocs, nthreads, maxmem, maxswp, and maxtmp are not static on UNIX hosts that support dynamic hardware reconfiguration.

Index	Measures	Units	Determined by
type	host type	string	configuration
model	host model	string	configuration
hname	host name	string	configuration
cpuf	CPU factor	relative	configuration
server	host can run remote jobs	Boolean	configuration
rexpri	execution priority	nice(2) argument	configuration
ncpus	number of processors	processors	LIM
ndiks	number of local disks	disks	LIM
maxmem	maximum RAM	MB	LIM
maxswp	maximum swap space	MB	LIM
maxtmp	maximum space in /tmp	MB	LIM

CPU factor (cpuf)

The CPU factor is the speed of the host's CPU relative to other hosts in the cluster. If one processor is twice the speed of another, its CPU factor should be twice as large. CPU factors are defined by the cluster administrator. For multiprocessor hosts, the CPU factor is the speed of a single processor.

Server

The server static resource is Boolean. It has the following values:

- 1 if the host is configured to run jobs from other hosts.
- 0 if the host is a client for submitting jobs to other hosts.

operator

The following operators can be used in selection strings. If you are using the selection string in an XML format, you must use the applicable escape characters in the XML Equivalence column. The operators are listed in order of decreasing precedence:

Operator	XML Equivalent	Syntax	Meaning
!	n/a	!a	Logical NOT: 1 if a==0, 0 otherwise
*	n/a	a*b	Multiply a and b
/	n/a	a / b	Divide a by b

Operator	XML Equivalent	Syntax	Meaning
+	n/a	a+b	Add a and b
-	n/a	a-b	Subtract b from a
>	>	a > b	1 if a is greater than b, 0 otherwise
<	<	a < b	1 if a is less than b, 0 otherwise
>=	>=	a >= b	1 if a is greater than or equal to b, 0 otherwise
<=	<=	a <= b	1 if a is less than or equal to b, 0 otherwise
==	n/a	a == b	1 if a is equal to b, 0 otherwise
!=	n/a	a != b	1 if a is not equal to b, 0 otherwise
&&	&&	a && b	Logical AND: 1 if both a and b are non-zero, 0 otherwise
	n/a	a b	Logical OR: 1 if either a or b is non-zero, 0 otherwise

value

Specifies the value to be used as criteria for selecting a resource. Value can be numerical, such as when referring to available memory or swap space, or it can be textual, such as when referring to a specific type of host.

Example: Simple selection expression using static resources

The following example selects resources with total memory greater than 500MB:

```
select (maxmem > 500)
```

Example: Compound selection expression using static resources

The following example selects resources with total memory greater than 500MB and total swap space greater than or equal to 300MB:

```
select (maxmem > 500 & & maxswp >= 300)
```

order

Sorts the selected resources into an order of preference according to the values of the resources.

order synopsis

order(*order_string*)

order description

The order string acts on the results of a select string, sorting the selected resources to identify the most favorable resources, and eliminate the least desirable resources.

Resources are sorted into ascending order based on the result of an arithmetical expression.

The result orders the resources from smallest to largest.

order_string

Specify an arithmetic expression that expresses the desired outcome.

You can create an expression using operators and numbers.

Use the following operators:

+	add
-	subtract-
*	multiply
/	divide

Order resources based on available swap space

The following example orders the selected resources based on their available swap space, but sorts by descending order, ordering hosts with the greatest amount of available swap space first:

```
order (0- swp)
```

resourceGroupName

Resource group from which resources are requested to run session managers. The resource group should be a management host resource group and must be an existing resource group in EGO.

Not required for Symphony DE.

Where used

- SOAM >SSM

Required/Optional

Optional

Default value

If the resource group is not defined, the system selects resources from the ManagementHosts group to start the session manager.

Related attributes

- resReq in SOAM > SSM—Specifies the criteria for selecting resources to start the session manager. Resources will only be selected from resource groups specified by resourceGroupName .

workDir

Absolute path in which the session manager process stores data for operations, including paging, journaling and history files.

Paging, journaling, and history data files are saved in subdirectories under the workDir if a relative path is configured for the paging, journaling, and data history paths.

This is not applicable in Symphony DE. In Symphony DE the working directory of the session manager is:

- Windows— %SOAM_HOME%\work
- Linux— \$SOAM_HOME/work

Where used

- SOAM > SSM

Required/Optional

Optional

Valid values

Any accessible absolute path on the network or local disk. To facilitate the session manager recovery, workDir should be set to a directory on a shared file system.

Default value

By default, the session manager working directory is the same for all applications. If applications are configured to write to the same directory, the operating system user accounts assigned to start the session manager must have access to that directory. It is recommended that all operating system execution users who start the session managers be in the same primary user group.

If a shared directory is configured in the cluster, the default working directory for the session manager points to the shared location in the host on which session manager runs:

- \${EGO_SHARED_TOP}/soam/work

If there is no shared directory configured in the cluster, then the default working directory points to the local location in the host on which session manager runs:

- Windows—%EGO_CONFDIR%\ . . \ . \soam\work
- Linux—\$EGO_CONFDIR/. . / . /soam/work

SSMResPubPluginCmd

Start-up command including path that launches the data-aware scheduling plug-in process. This process is used in conjunction with an external application to request the location and cost values for preferred user-defined service attributes.

Parameter SSMResPubPluginCmd supports substitution with the following Symphony variables:

- SOAM_HOME
- VERSION_NUM
- EGO_CONFDIR
- EGO_SHARED_TOP
- EGO_MACHINE_TYPE

For example:

```
SSMResPubPluginCmd="${SOAM_HOME}/${VERSION_NUM}/${EGO_MACHINE_TYPE}/bin/MyPlugin"
```


SOAM section

Where used

- SOAM > SSM

Required/Optional

Optional

Valid values

Any location accessible on your network or local disk.

SSMResPubCacheEnabled

Specifies whether caching is enabled for results received from the plug-in so that subsequent calls for the same attributes will return cached values.

Where used

- SOAM > SSM

Required/Optional

Optional

Valid values

true | false

Default value

true

SOAM>SSM>boundaryManagerConfig>elementDefinitions>boundaryElement

For internal system use only.

name

For internal system use only.

description

For internal system use only.

minValue

For internal system use only.

maxValue

For internal system use only.

SOAM>SSM>boundaryManagerConfig>elementDefinitions>boundaryElement>additionalConfig

For internal system use only.

name

For internal system use only.

value

For internal system use only.

SOAM >SSM > boundaryManagerConfig>boundaries>boundary

elementName

Specifies how the system responds to changes in the amount of physical memory, virtual memory, or virtual address space on the session manager host.

- **AvailableMemory**—The boundary manager monitors the amount of virtual memory available on the host. Virtual memory is physical memory plus swap memory that is available on the host.
- **AvailableVirtualAddressSpace**—The boundary manager monitors the amount of virtual address space the session manager can use to map memory. It is possible for the host to have more than enough memory available, which is physical memory plus swap memory, but the process cannot access anymore because it has no available address space to map this memory to.
- **ProcessMemory**—The boundary manager monitors the amount of physical memory that is available for each process.

Where used

SOAM > SSM > boundaryManagerConfig > boundaries > boundary

Required/Optional

Optional

Valid values

AvailableMemory | AvailableVirtualAddressSpace | ProcessMemory

Related attributes

- **name (event)**—Multiple boundary events are associated with each elementName
- **value (event)**—A value is assigned to each boundary event
- **name (param)**—The MaxSizeReference parameter is associated with AvailableVirtualAddressSpace and ProcessMemory
- **value (param)**—A value is assigned to the MaxSizeReference parameter

SOAM >SSM > boundaryManagerConfig>boundaries>boundary>event

name (event)

Specifies the name of the boundary event that triggers a system response when the value of the event is reached.

The system monitors the boundaries defined, available memory, and available virtual address space, to ensure the robust functionality of the session manager. Messages are logged to the session manager log file and administrators are notified through event notification when new event are triggered..

Memory low conditions can be caused by:

- Running too many memory -consuming processes on the session manager host
- Too many unprocessed input and output messages
- Too many unfinished sessions or tasks

The following describes system reactions when a boundary limit is reached:

Boundary event	Messages	Workload processing changes	System processing changes
BEV_PROACTIVE	INFO	The SSM starts to page data, which can lead to slower input handling. The impact is dependent on message size, disk speed, etc.	Starts scheduling common data and input/output messages for paging.
BEV_SEVERE	WARNING	<ul style="list-style-type: none"> • The SSM starts to release already paged data from memory. This will impact the time to dispatch tasks as they must be read back into memory before being dispatched. • The session manager allows clients to create new sessions and submit new tasks. These actions were prohibited by the session manager after the BEV_CRITICAL event was triggered. 	Assigns higher priority to paging write requests, memory is released once the messages are paged out
BEV_CRITICAL	WARNING	<ul style="list-style-type: none"> • New connections from the client to the session manager are rejected • New session creations and tasks submissions are held until a lower level event is triggered such as BEV_SEVERE or BEV_PROACTIVE 	Assigns higher priority to paging write requests
BEV_HALT	FATAL	<ul style="list-style-type: none"> • All new session and client requests are rejected • Processing of existing requests is stopped 	<ul style="list-style-type: none"> • Runs in maintenance mode • Only cluster administrator can access workload through the console • Once memory drops down, and BEV_SEVERE or BEV_PROACTIVE is triggered, the session manager resumes work

Memory low conditions can be corrected by:

- Running the session manager on a dedicated management host
- Adding more physical memory or increasing the amount of assigned swap memory
- Terminating any faulty sessions that are consuming excessive amounts of memory
- Terminating any nominal sessions to release memory

Where used

SOAM > SSM > boundaryManagerConfig > boundaries > boundary > event

Required/Optional

Optional

Valid values

BEV_PROACTIVE | BEV_SEVERE | BEV_CRITICAL | BEV_HALT

Related attributes

- **elementName**—The name of the system resource monitored by the boundary manager
- **value (event)**—The minimum value the system resource can reach before the event is triggered

value (event)

Specifies the value of a boundary event as a percentage of the entire resource available.

When the availability of the resource monitored falls below the boundary event value, it triggers a system response to ensure the session manager has enough memory to function.

Where used

SOAM > SSM > boundaryManagerConfig > boundaries > boundary > event

Required/optional

Optional

Valid values

0 or more

Default values

The following describes the element names, event names, and their default boundary values:

elementName / event name	BEV_PROACTIVE	BEV_SEVERE	BEV_CRITICAL	BEV_HALT
AvailableMemory	50	40	25	15
AvailableVirtualAddressSpace	50	40	25	15
ProcessMemory	0	0	0	0

Related attributes

- **elementName**—Specifies the resource to which the boundary applies.
- **name (event)**—Specifies the boundary event for which a value is assigned.

SOAM >SSM > boundaryManagerConfig>boundaries>boundary>param

name (param)

Specifies the name of the boundary parameter. The boundary parameter and its associated value are used as a reference against which boundary events are triggered for AvailableVirtualAddressSpace and ProcessMemory.

The system monitors the boundaries defined to ensure the robust functionality of the session manager. Messages are logged to the session manager log file and administrators are notified through event notification when new events are triggered.

For a description of system reactions when boundary events occur, refer to the description for SOAM > SSM > boundaryManagerConfig > boundaries > boundary > event.

Where used

SOAM > SSM > boundaryManagerConfig > boundaries > boundary > param

Required/Optional

Optional

Valid values

MaxSizeReference

Related attributes

- elementName—The name of the system resource monitored by the boundary manager
- value (param)—The minimum value the system resource can reach before the event is triggered

value (param)

The value is used in conjunction with events BEV_PROACTIVE, BEV_SEVERE, BEV_CRITICAL, BEV_HALT. When the value of MaxSizeReference is defined, the percentage associated with the boundary events is a ratio based on this value instead of the default value.

- For element AvailableVirtualAddressSpace, the value of MaxSizeReference defines the maximum virtual address space in MB. If the value of MaxSizeReference is not defined or is not valid, the default value of 2G (for 32 bit OS) and 8T (for 64 bit OS) is used.
- For element "ProcessMemory", the value of MaxSizeReference defines the quota of the memory size for the SSM in MB. If the value of MaxSizeReference is not defined or is not valid, the default value of 2G (for 32 bit OS) and 8T (for 64 bit OS) is used.

Where used

SOAM > SSM > boundaryManagerConfig > boundaries > boundary > param

Required/optional

Optional

Valid values

0 or more; if 0 is defined, the default value is used (8T for 64 bit OS and 2G for 32 bit OS)

Related attributes

- elementName—Specifies the resource (ProcessMemory or AvailableVirtualAddressSpace) to which the boundary applies.
- name (param)—Specifies the boundary for which a value is assigned.

SOAM>SSM>boundaryManagerConfig>globalConfig

For internal system use only.

name

For internal system use only.

value

For internal system use only.

SOAM>SIM

blockHostOnTimeout

Used with the startUpTimeout (SOAM > SIM) attribute. Defines whether to block this host for the application when the service instance manager times out on process startup. If set to true, when the service instance manager times out on startup, the host is added to the blocked host list and is no longer used for the application unless the host is explicitly unblocked through the Platform Management Console.

Symphony DE blocks slots rather than hosts. Slots can be unblocked by disabling and enabling your application.

Where used

SOAM > SIM

Required/Optional

Optional

Valid values

true | false

Default value

true, host is added to the blocked host list for the application upon timeout

Related attributes

startUpTimeout (SOAM > SIM)

blockHostOnVersionMismatch

For internal system use only.

startUpTimeout

Number of seconds to wait for the service instance manager to start up before the session manager considers it as timed out. This parameter works in conjunction with blockHostOnTimeout.

After a session manager starts a service instance manager, if the service instance manager cannot contact the session manager within the startUpTimeout, the session manager requests a new host from EGO and tries to start a new service instance manager on the new host.

Where used

SOAM > SIM

Required/Optional

Optional

Valid values

1 or more

Default value

120 seconds

Related attributes

- `blockHostOnTimeout`—Determines whether the host is blocked if timeout occurs on SIM startup.

SOAM>DataHistory

Parameters related to sessions and tasks historical data storage.

pollFrequency

Specifies the frequency, in seconds, at which the history files are polled to determine if the current file should be archived.

Where used

SOAM > DataHistory

Required/Optional

Optional

Valid values

1 to 10 (inclusively)

Default value

10 seconds

Related attributes

- `fileSwitchSize`—Defines, according to file size, when old history files are archived and new history files are started
- `fileSwitchTime`—Defines, according to file age, when old history files are archived and new history files are started

fileSwitchSize

Maximum size, in megabytes, the history file can reach before the file is archived at the next polling interval.

When the history file reaches the maximum size specified, the file is renamed by appending a time stamp to the file name.

Data history file names have the following format:

applianceName_session.soamdb or *applianceName_task.soamdb*

After renaming the file names would be similar to the following:

applianceName_session.soamdb.yyyy_mm_dd_hh_mm_ss_x

applianceName_task.soamdb.yyyy_mm_dd_hh_mm_ss_x

Where used

SOAM > DataHistory

Required/Optional

Optional

Valid values

1 to 100 (inclusively)

Default value

100 MB

Related attributes

pollFrequency—The size of the file is checked at the time interval defined by pollFrequency

fileSwitchTime

Maximum number of hours that can elapse before the history file is archived at the next polling interval.

When the history file reaches the maximum age specified, the file is renamed by appending a time stamp to the file name.

Data history file names have the following format:

applicationName_session. soamdb or *applicationName_task. soamdb*

After renaming the file names would be similar to the following:

applicationName_session. soamdb. yyyy_mm_dd_hh_mm_ss_x

applicationName_task. soamdb. yyyy_mm_dd_hh_mm_ss_x

Where used

SOAM > DataHistory

Required/Optional

Optional

Valid values

1 to 168(1 week)

Default value

24 hours

Related attributes

pollFrequency—The age of the file is checked at the time interval defined by pollFrequency

lastingPeriod

Number of hours the archived history files are retained before deleted.

Where used

SOAM > DataHistory

Required or optional

Optional

Range or valid values

1 to 168 (1 week)

Default value

96 hours (4 days)

lastingPeriodInSeconds

For internal system use only.

path

Absolute or relative path in which session and task historical files are stored. If relative path is specified, the directory is located under session manager working directory. The path should be accessible by both session director and session manager, so that the session and task historical data can be retrieved by command line and PMC.

Note:

Ensure the user account running the session director (cluster administrator), and the user account running the session manager for the application (os execution account assigned to the consumer) has access to the directory.

By default, history files are saved under the following locations:

- For Symphony,
 - Windows—%EGO_CONFDIR%\.. \.. \soam\work\hi story
 - Linux—\$EGO_CONFDIR/.. /.. /soam/work/hi story
- For Symphony DE,
 - Windows—%SOAM_HOME%\work\hi story
 - Linux—\$SOAM_HOME/work/hi story

If you specify an absolute path, all the history files are saved in the specified path. An absolute path must be accessible by both session director and the session manager for this application.

Where used

- SOAM > DataHistory

Required/Optional

Optional

Valid values

Any absolute or relative path on network or local disk which can be accessed by both session director and session manager.

Default value

By default, data history files are saved in the "history" directory under the session manager working directory.

Related attributes

- workDir (SOAM > SSM)—the working directory of session manager.

delimiter

For internal system use only.

SOAM>PagingTasksInput

Parameters related to task input message paging configurations.

path

Directory where paging files for task input are stored. The path to the file can be specified as a relative or absolute path. If a relative path is specified, paging files are saved in a subdirectory relative to the session manager working directory. The specified directory must be accessible from all management hosts to facilitate session manager failover.

Where used

- SOAM > PagingTasksInput

Required/Optional

Optional

Valid values

Any accessible absolute or relative path on network or local disk that can be accessed from all management hosts.

Default values

By default, paging files are saved in the "paging" directory under the session manager working directory.

Related attributes

- workDir (SOAM > SSM)—the working directory of session manager.

blockSize

Specifies the number of bytes per block for paging. The block size should be large enough to contain the largest message. If the block size is not large enough, the data is spread across multiple blocks, which can slow down paging processes.

Consider the size of the largest input message when determining the optimal blocksize for paging input tasks

Where used

- SOAM > PagingTasksInput

SOAM section

Required/Optional

Optional

Valid values

1 or more

Default value

4096 bytes (4KB)

diskSpace

Maximum amount of disk space, in bytes, dedicated to paging task input messages.

Where used

SOAM > PagingTasksInput

Required/Optional

Optional

Valid values

1 or more

Default values

4294967296 bytes (4 GB)

bitmapBits

For internal system use only.

pmeType

For internal system use only.

SOAM>PagingTasksInput>PagingMode

For internal system use only.

name

For internal system use only.

readPriority

For internal system use only.

writePriority

For internal system use only.

removePriority

For internal system use only.

SOAM>PagingTasksInputNonRec

Parameters related to task input message paging configurations for non- recoverable sessions.

Note:

The PagingTasksInputNonRec and PagingTasksOutputNonRec elements must be set in pairs in the application profile; if only one of them is configured, it is ignored by the system.

path

Directory where paging files for task input are stored. The path to the file can be specified as a relative or absolute path. If a relative path is specified, paging files are saved in a subdirectory relative to the session manager working directory. To speed up paging and session manager recovery, the specified directory can be on the local drive since the paged data of non-recoverable sessions does not need to be persisted at a shared location.

Where used

- SOAM > PagingTasksInputNonRec

Required/Optional

Optional

Valid values

Any accessible absolute or relative path on network or local disk.

Default values

By default, paging files are saved in the "paging" directory under the session manager working directory.

Related attributes

- workDir (SOAM > SSM)—the working directory of session manager.

blockSize

Specifies the number of bytes per block for paging. The block size should be large enough to contain the largest message. If the block size is not large enough, the data is spread across multiple blocks, which can slow down paging processes.

Consider the size of the largest input message when determining the optimal blocksize for paging input tasks

Where used

- SOAM > PagingTasksInputNonRec

Required/Optional

Optional

SOAM section

Valid values

1 or more

Default value

4096 bytes (4KB)

diskSpace

Maximum amount of disk space, in bytes, dedicated to paging task input messages.

Where used

SOAM > PagingTasksInputNonRec

Required/Optional

Optional

Valid values

1 or more

Default values

4294967296 bytes (4 GB)

bitmapBits

For internal system use only.

pmeType

For internal system use only.

SOAM>PagingTasksInputNonRec>PagingMode

For internal system use only.

name

For internal system use only.

readPriority

For internal system use only.

writePriority

For internal system use only.

removePriority

For internal system use only.

SOAM>PagingTasksOutput

Parameters related to task output message paging configurations.

path

Directory where paging files for task output are stored. The path to the files can be specified as a relative or absolute path. If a relative path is specified, paging files are saved in a subdirectory relative to the session manager working directory. The specified directory must be accessible from all management hosts to facilitate session manager failover.

Where used

- SOAM > PagingTasksOutput

Required/Optional

Optional

Valid values

Data files can be stored on any storage device accessible on the network. To facilitate failover, Symphony working files should be stored on a shared file system.

Default values

By default, paging files are saved in the "paging" directory under the session manager working directory.

Related attributes

- workDir (SOAM > SSM)—the working directory of session manager.

blockSize

Specifies the number of bytes per block for paging. The block size should be large enough to contain the largest message. If the block size is not large enough, the data is spread across multiple blocks, which can slow down paging processes.

Consider the size of the largest output message when determining the optimal blocksize for paging output tasks

Where used

- SOAM > PagingTasksOutput

Required/Optional

Optional

Valid values

1 or more

Default value

4096 bytes (4KB)

bitmapBits

For internal system use only.

diskSpace

Maximum amount of disk space, in bytes, dedicated to paging task output messages.

SOAM section

Where used

SOAM > PagingTasksOutput

Required/Optional

Optional

Valid values

1 or more

Default value

4294967296 bytes (4 GB)

pmeType

For internal system use only.

SOAM>PagingTasksOutput>PagingMode

For internal system use only.

name

For internal system use only.

readPriority

For internal system use only.

writePriority

For internal system use only.

removePriority

For internal system use only.

SOAM>PagingTasksOutputNonRec

Parameters related to task output message paging configurations for non-recoverable sessions.

Note:

The PagingTasksInputNonRec and PagingTasksOutputNonRec elements must be set in pairs in the application profile; if only one of them is configured, it is ignored by the system.

path

Directory where paging files for task output are stored. The path to the files can be specified as a relative or absolute path. If a relative path is specified, paging files are saved in a subdirectory relative to the session manager working directory. To speed up paging and session manager recovery, the specified directory can be on the local drive since the paged data of non-recoverable sessions does not need to be persisted at a shared location.

Where used

- SOAM > PagingTasksOutputNonRec

Required/Optional

Optional

Valid values

Any accessible absolute or relative path on network or local disk.

Default values

By default, paging files are saved in the "paging" directory under the session manager working directory.

Related attributes

- workDir (SOAM > SSM)—the working directory of session manager.

blockSize

Specifies the number of bytes per block for paging. The block size should be large enough to contain the largest message. If the block size is not large enough, the data is spread across multiple blocks, which can slow down paging processes.

Consider the size of the largest output message when determining the optimal blocksize for paging output tasks

Where used

- SOAM > PagingTasksOutputNonRec

Required/Optional

Optional

Valid values

1 or more

Default value

4096 bytes (4KB)

diskSpace

Maximum amount of disk space, in bytes, dedicated to paging task output messages.

Where used

SOAM > PagingTasksOutputNonRec

Required/Optional

Optional

Valid values

1 or more

SOAM section

Default value

4294967296 bytes (4 GB)

bitmapBits

For internal system use only.

pmeType

For internal system use only.

SOAM>PagingTasksOutputNonRec>PagingMode

For internal system use only.

name

For internal system use only.

readPriority

For internal system use only.

writePriority

For internal system use only.

removePriority

For internal system use only.

SOAM>PagingCommonData

Parameters related to common data paging configurations.

path

Directory where the files are stored for common data. The path to the files can be specified as a relative or absolute path. If a relative path is specified, common data files are saved in a subdirectory relative to the session manager working directory. The specified directory must be accessible from all management hosts to facilitate session manager failover.

Where used

- SOAM > PagingCommonData

Required/Optional

Optional

Valid values

Data files can be stored on any storage device accessible on the network. To facilitate failover, Symphony working files should be stored on a shared file system.

Default values

By default, paging files are saved in the "paging" directory under the session manager working directory.

Related attributes

- workDir (SOAM > SSM)—the working directory of session manager.

blockSize

Specifies the number of bytes per block for paging. The block size should be large enough to contain the largest message. If the block size is not large enough, the data is spread across multiple blocks, which can slow down paging processes.

Consider the size of the largest common data message when determining the optimal block size for paging common data.

Where used

- SOAM > PagingCommonData

Required/Optional

Optional

Valid values

1 or more

Default value

102400 bytes (100KB)

diskSpace

Specifies the maximum amount of disk space, in bytes, dedicated to paging.

Where used

SOAM > PagingCommonData

Required/Optional

Optional

Valid values

1 or more

Default values

8589934592 bytes (8GB)

bitmapBits

For internal system use only.

pmeType

For internal system use only.

SOAM>PagingCommonData>PagingMode

For internal system use only.

name

For internal system use only.

readPriority

For internal system use only.

writePriority

For internal system use only.

removePriority

For internal system use only.

SOAM>PagingCommonDataNonRec

Parameters related to common data paging configurations for non-recoverable sessions.

Note:

The PagingCommonDataNonRec and PagingCommonDataUpdatesNonRec elements must be set in pairs in the application profile; if only one of them is configured, it is ignored by the system.

path

Directory where the files are stored for common data. The path to the files can be specified as a relative or absolute path. If a relative path is specified, common data files are saved in a subdirectory relative to the session manager working directory. To speed up paging and session manager recovery, the specified directory can be on the local drive since the paged data of non-recoverable sessions does not need to be persisted at a shared location.

Where used

- SOAM > PagingCommonDataNonRec

Required/Optional

Optional

Valid values

Any accessible absolute or relative path on network or local disk.

Default values

By default, paging files are saved in the "paging" directory under the session manager working directory.

Related attributes

- workDir (SOAM > SSM)—the working directory of session manager.

blockSize

Specifies the number of bytes per block for paging. The block size should be large enough to contain the largest message. If the block size is not large enough, the data is spread across multiple blocks, which can slow down paging processes.

Consider the size of the largest common data message when determining the optimal block size for paging common data.

Where used

- SOAM > PagingCommonDataNonRec

Required/Optional

Optional

Valid values

1 or more

Default value

102400 bytes (100KB)

diskSpace

Specifies the maximum amount of disk space, in bytes, dedicated to paging.

Where used

SOAM > PagingCommonDataNonRec

Required/Optional

Optional

Valid values

1 or more

Default values

8589934592 bytes (8GB)

bitmapBits

For internal system use only.

pmeType

For internal system use only.

SOAM>PagingCommonDataNonRec>PagingMode

For internal system use only.

name

For internal system use only.

readPriority

For internal system use only.

writePriority

For internal system use only.

removePriority

For internal system use only.

SOAM>PagingCommonDataUpdates

Parameters related to common data updates paging configurations.

path

Directory where the files are stored for common data updates. The path to the files can be specified as a relative or absolute path. If a relative path is specified, common data update files are saved in a subdirectory relative to the session manager working directory. The specified directory must be accessible from all management hosts to facilitate session manager failover.

Where used

- SOAM > PagingCommonDataUpdates

Required/Optional

Optional

Valid values

Data files can be stored on any storage device accessible on the network. To facilitate failover, Symphony working files should be stored on a shared file system.

Default values

By default, paging files are saved in the "paging" directory under the session manager working directory.

Related attributes

- workDir (SOAM > SSM)—the working directory of session manager.

blockSize

Specifies the number of bytes per block for paging. The block size should be large enough to contain the largest message. If the block size is not large enough, the data is spread across multiple blocks, which can slow down paging processes.

Consider the size of the largest common data update message when determining the optimal block size for paging common data updates.

Where used

- SOAM > PagingCommonDataUpdates

Required/Optional

Optional

Valid values

1 or more

Default value

102400 bytes (100KB)

diskSpace

Specifies the maximum amount of disk space, in bytes, dedicated to paging.

Where used

SOAM > PagingCommonDataUpdates

Required/Optional

Optional

Valid values

1 or more

Default values

8589934592 bytes (8GB)

bitmapBits

For internal system use only.

pmeType

For internal system use only.

SOAM>PagingCommonDataUpdates>PagingMode

For internal system use only.

name

For internal system use only.

readPriority

For internal system use only.

writePriority

For internal system use only.

removePriority

For internal system use only.

SOAM>PagingCommonDataUpdatesNonRec

Parameters related to common data updates paging configurations for non-recoverable sessions.

Note:

The PagingCommonDataNonRec and PagingCommonDataUpdatesNonRec elements must be set in pairs in the application profile; if only one of them is configured, it is ignored by the system.

path

Directory where the files are stored for common data updates. The path to the files can be specified as a relative or absolute path. If a relative path is specified, common data update files are saved in a subdirectory relative to the session manager working directory. To speed up paging and session manager recovery, the specified directory can be on the local drive since the paged data of non-recoverable sessions does not need to be persisted at a shared location.

Where used

- SOAM > PagingCommonDataUpdatesNonRec

Required/Optional

Optional

Valid values

Any accessible absolute or relative path on network or local disk.

Default values

By default, paging files are saved in the "paging" directory under the session manager working directory.

Related attributes

- workDir (SOAM > SSM)—the working directory of session manager.

blockSize

Specifies the number of bytes per block for paging. The block size should be large enough to contain the largest message. If the block size is not large enough, the data is spread across multiple blocks, which can slow down paging processes.

Consider the size of the largest common data update message when determining the optimal block size for paging common data updates.

Where used

- SOAM > PagingCommonDataUpdatesNonRec

Required/Optional

Optional

Valid values

1 or more

Default value

102400 bytes (100KB)

diskSpace

Specifies the maximum amount of disk space, in bytes, dedicated to paging.

Where used

SOAM > PagingCommonDataUpdateNonRec

Required/Optional

Optional

Valid values

1 or more

Default values

8589934592 bytes (8GB)

bitmapBits

For internal system use only.

pmeType

For internal system use only.

SOAM>PagingCommonDataUpdatesNonRec>PagingMode

For internal system use only.

name

For internal system use only.

readPriority

For internal system use only.

writePriority

For internal system use only.

removePriority

For internal system use only.

SOAM>JournalingTasks

path

For internal system use only.

blockSize

For internal system use only.

bitmapBits

For internal system use only.

diskSpace

For internal system use only.

pmeType

For internal system use only.

SOAM>JournalingSessions

path

For internal system use only.

blockSize

For internal system use only.

bitmapBits

For internal system use only.

diskSpace

For internal system use only.

pmeType

For internal system use only.

SOAM>JournalingSessionTagConfig

path

For internal system use only.

blockSize

For internal system use only.

bitmapBits

For internal system use only.

diskSpace

For internal system use only.

pmeType

For internal system use only.

SessionTypes section

A session type defines a group of configuration settings for session attributes. Multiple session types can be defined in this section with different configuration settings for each session type. Symphony also defines a system default session type with the name "". You can add or update (or remove) a session type to an existing application dynamically using PMC.

A client application specifies the session type when creating a session. The session type that the client specifies must be configured in this section or it should use the system default.

Type

The Type element defines a session type, which is a group of configuration settings for session attributes.

name

Session type name is an identifier for the group of configuration settings. A client application specifies the session type name when creating a session. When a session is created, the configuration settings for the specified session type are applied to that session.

The session type is optional. If you leave this parameter blank "" or do not set a session type, system default values are used for session attributes. If you specify a session type in the client application, you must also configure the session type in the application profile—the session type name in your application profile and session type you specify in the client must match. If you use an incorrect session type in the client and the specified session type cannot be found in the application profile, an exception is thrown to the client.

Where used

SessionTypes > Type

Required/Optional

Required

Valid values

1—256 alphanumeric characters, including all printable characters.

serviceName

Name of service that is used to execute the workload for the sessions in this session type. This is any name that you assign to the service to link the session type to the service definition. The name you enter here must match the name entered in the <Service> section.

Specify a service name only when multiple services are defined in the application profile and when you want to assign the session type a different service than the default service. In the client application you can use the appropriate API to override the service that is used to execute the workload for the session that you are creating. Refer to the API Reference for more details.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

1—256 alphanumeric characters, including all printable characters.

Default value

If only one service section is defined in the application profile, uses the service defined in the Service section by default. If more than one service is defined in the application profile, uses the service identified as the default service by default.

priority

Specifies the session priority, where 1 is the lowest priority and 10000 is the highest priority. If there are multiple open sessions, the session priority affects how resources are distributed across sessions.

The resource distribution is also affected by the scheduling policy of the session manager. For more details, refer to the `policy` attribute in the Consumer section.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

1 to 10000

Default value

1

Related attributes

policy—Determines how resources are allocated to sessions according to the scheduling policy used

preemptive

Specifies whether the session can preempt other sessions when it has workload to run.

If true, the session preempts sessions to get resources to finish the session as soon as possible.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

true | false

Default value

false

Related attributes

- **policy**—Determines how resources are allocated to sessions according to the scheduling policy used.
- **preemptionScope**—Determines whether sessions in the application can preempt sessions of lower or equal rank, or only preempt sessions of lower rank.

minServices

Minimum number of CPU slots required for each open session of this type. The minServices attribute is applicable only when the R_MinimumServices scheduling policy is used.

Symphony attempts to allocate the minimum required number of CPU slots to each session before proportionally allocating additional available CPUs based on

Service instances started on the minimum number of CPU slots remain bound to the session until the session is suspended, killed or closed.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

0-10000

Default value

0

Related attributes

policy—Determines how resources are allocated to sessions according to the scheduling policy used. The minServices attribute is applicable only when the R_MinimumServices scheduling policy is used.

recoverable

Specifies whether the session and its workload can be recovered if the session manager restarts or fails over.

If true, the data for tasks and other session information are retained to attempt to recover the session in a failover situation.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

true | false

Default value

false

Related attributes

- `workDir` (SOAM > SSM) —Defines where working files are stored

sessionRetryLimit

Specifies the number of times the session manager retries to bind the service to a session if the service fails in the `SessionEnter` or `SessionUpdate` API method. After the session experiences the specified number of retries, the session is aborted.

Note:

The retry count for both of these methods are considered together. For example, if `SessionEnter` fails once and `SessionUpdate` fails twice, then the session rerun count is equal to 3. Therefore the `SessionRetryCount` should be set to a value that accounts for both `SessionEnter` and `SessionUpdate` failures.

You can specify what events you want Symphony to consider as bind failures by configuring attributes in the `Service > Control > Method` section for your service.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

0 or more

Default value

3

taskRetryLimit

Specifies the number of attempts to retry a task in the session if the task does not complete successfully. After a task has reached this number of retries and still does not complete successfully, the task is put into the Error state.

This attribute is used if the service fails during the API `Invoke` method.

If the value of `taskRetryLimit` is 3, the system makes a maximum of 4 attempts to run the task before the task fails.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

0 or more

Default value

3

taskGracePeriod

Number of milliseconds to allow a task to continue running after a session of the task is suspended by a user.

Number of milliseconds to allow the running service method to complete and to allow the service instance to clean up after the resource on which the service instance is running is reclaimed.

For clean up, Symphony will initiate the onSessionLeave() method and the onDestroyService() (if applicable) after the current running service method completes.

- If a task is running and the Invoke method completes during the task grace period, the result of that method is treated as it would be treated under normal circumstances.
- If a task is running and the Invoke method does not complete before the task grace period expires, the service instance on which the task is running is terminated and the task is requested.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

0 or more

Default value

100 milliseconds

Related attributes

- suspendGracePeriod - If suspendGracePeriod is defined in the application profile, it overrides the value for taskGracePeriod for the suspend scenario.
- reclaimGracePeriod - If reclaimGracePeriod is defined in the application profile, it overrides the value for taskGracePeriod for the reclaim scenario.

reclaimGracePeriod

Number of milliseconds to allow the current running service method to complete and the service instance to clean up when the resource on which the service instance is running is reclaimed. If the service method and cleanup does not complete within the set time, then Symphony will terminate the instance. If the timeout has not expired, Symphony will initiate cleanup after the current running service method completes.

When this reclaimGracePeriod < EGO Consumer reclaim grace period, the reclaimGracePeriod takes effect.

When this reclaimGracePeriod >= EGO Consumer reclaim grace period, the EGO Consumer reclaim grace period takes effect.

If the Invoke method was running and completes during the applied reclaim grace period, the result of that method is treated as it would be treated under normal circumstances.

If the Invoke method was running and does not complete before the applied reclaim grace period expires, the service instance on which the task is running is terminated and the task is requeued.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

- 0 or more
- -1 indicates unlimited grace period, the EGO Consumer reclaim grace period is used instead

Default value

-1

suspendGracePeriod

Number of milliseconds to allow a task to continue running after the session for the task is suspended by a user.

A task is considered to be running on a service instance when that service instance is executing its Invoke method for that particular task.

- If the Invoke method completes during the suspend grace period, the result of that method is treated as it would be treated under normal circumstances.
- If the Invoke method does not complete before the suspend grace period expires, the service instance on which the task is running is terminated and the task is requeued.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

0 or more

Default value

100 milliseconds

taskCleanupPeriod

Number of milliseconds to allow a running task to finish executing when the corresponding session has been aborted by the system or a user, or a task is killed by a user. When a session is aborted, the running tasks of the session are canceled, and the service instance manager will not return any task output from the service instance to the session manager even if the task completes before the taskCleanupPeriod expires.

The service instances which have running tasks will not be released until the `taskCleanupPeriod` expires or the `APIInvoke` call returns.

Where used

SessionTypes

Required /Optional

Optional

Valid values

0 or more

Default value

250 milliseconds

abortSessionIfClientDisconnect

Specifies whether the session is aborted if the session manager detects that the connection between the client and the session manager is broken.

- If true, and the `transientDisconnectionTimeout` expires, the session is aborted.
- If false, and the `transientDisconnectionTimeout` expires, the session is retained and its workload continues to run. The client can reconnect to session to resume its workload.

Note:

Note that if in a new connection a session that was previously disconnected is opened within the `transientDisconnectionTimeout` period after the original client exited abnormally, the session is not aborted even if `abortSessionIfClientDisconnect` is set to true.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

true | false

Default value

true

Related attributes

`transientDisconnectionTimeout`—Defines how long the session manager waits for a client to reconnect before it aborts the sessions that are open on the client's connection.

abortSessionIfTaskFail

Specifies whether the session is aborted if a task fails (enters the Error state).

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

true | false

Default value

false

persistSessionHistory

Specifies whether to persist session history. If any form of task history is retained, session history must also be retained.

Session history information includes, but is not limited to:

- Session creation and end times
- Final state reached by the session
- Summary of tasks for the session

Any unrecoverable sessions that are not in their final states are not recorded.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

none | all

Default value

all

Related attributes

- path(SOAM > DataHistory)—Session history is stored in the history files whose location is defined by path
- persistTaskHistory—If task history is retained, session history must also be retained

persistTaskHistory

Specifies whether to retain no task history, all task history, or only task history for error tasks. Session history must also be retained for task history to be retained.

- none—Does not persist task history

- all—Persists history for all tasks
- error—Persists history only for tasks that end in error

The task history contains details about the task such as run time, rerun, and rerun counts.

Any unrecoverable tasks that are not in their final states are not recorded.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

none | all | error

Default value

all

Related attributes

- path (SOAM > DataHistory)—Task history is stored in the task history files whose location is defined by path
- persistSessionHistory—If task history is retained, session history must also be retained

discardResultsOnDelivery

Specifies whether the system discards task output for the session when output has been retrieved by the client.

A value of true sets data to be discarded once output has been retrieved by the client.

A value of false sets data to be discarded only after a session is closed or aborted. Note that if set to false, outstanding output for all sessions are tracked until the session completes.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

true | false

Default value

true

serviceToSlotRatio

The number of slots required to run a service instance, represented by a service to slot ratio. Tasks from a session can only run on service instances that occupy the appropriate number of slots, determined by this ratio.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

- 1:N, where N is a positive integer. A 1:N ratio means that 1 service instance runs on multiple slots.
- N:1, where N is a positive integer with a maximum value of 10. An N:1 ratio means that multiple service instances run on 1 slot.

Default value

1:1

resourceGroupFilter

A list of resource groups from which the session can use resources. Tasks from a session can only run on resources that belong to one of the resource groups listed in the filter. The value of resourceGroupFilter must match or be a subset of the resourceGroupName specified in the Consumer section of the application profile.

Where used

SessionTypes > Type

Required/Optional

Optional

Default value

" "(empty)— resource groups specified at the application level (Consumer section) will be used.

Related attributes

resourceGroupName— Resource group from which resources are requested to run service instance managers and services to perform computations for clients.

preemptionRank

Preemption rank is only used with proportional or minimum services scheduling policy. It indicates the relative importance of a session during the resource reclaim process. Preemption rank is ignored if the priority scheduling policy is in use; in this case, session priority determines the reclaim order.

When the system needs to reclaim a resource, preemption rank is used to determine the relative importance of different sessions, so the system can reclaim a resource from the least important session.

By default, sessions are assigned the lowest rank, 1.

To help protect more important sessions from having a resource reclaimed, assign a higher rank to those sessions only. In a case where child tasks cannot complete after the parent is interrupted, it is a good idea to let the child tasks have a lower session rank, and assign a higher rank to the parent task.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

1-10000

Default value

1

enableCommonDataOptimization

Determines whether common data optimization for multi-slot hosts is enabled. When enabled, common data optimization results in the SSM sending only one copy of common data and common data updates to a multi-slot host allocated to the session. The common data and common data updates are shared by all SIMs serving the session on the same host.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

true | false

Default value

false

preference

Resource preference for the session expressed as a string. The expression, which contains resource attributes capable of being monitored, is evaluated against the attributes of each service/host available to the session in order to find the most preferred resource for the session.

Where used

SessionTypes > Type

Required/Optional

Optional

SessionTypes section

Valid values

Accepts alphanumeric characters and underscore and must start with an alphabetical character. Valid operators include +, -, *, and /.

defaultResourceAttributeValue

The default value that is substituted for the term in the preference expression when the value for the term is unknown.

Where used

SessionTypes > Type

Required/Optional

Optional

Valid values

-1.0E+300 to 1.0E+300

Default value

1.0E+300

Service section

The Service element defines a group of configuration settings for service attributes. Multiple service elements can be defined in this section with different configuration settings for each service. Defining multiple services allows you to use more than one service within a single application. You can add or update (or remove) a service to an existing application dynamically using PMC.

name

Name of the service to run for this application. The service name must be unique within the application.

A client application can use the appropriate API to override the service that is used to execute workload. Refer to the API Reference for more details. The service name that the client specifies must be configured in the application profile.

Where used

Service

Required/Optional

Required

Valid values

1—256 alphanumeric characters, including all printable characters.

Related attributes

- `serviceName`—Name of service that is used to execute the workload for the sessions in this session type.

description

String that describes the service.

Where used

Service

Required/Optional

Optional

Valid values

1—256 alphanumeric characters, including all printable characters.

default

Identifies a service as the default service, which means that the default service is immediately started when the service instance manager starts.

Where used

Service

Required/Optional

Optional when one service section is defined.

Required when multiple service sections are defined.

Valid values

true | false

Default value

True if only one service is defined in the application profile. If only one service is defined, that service is identified as the default and is started when the service instance manager starts.

When multiple services are defined, at least one of the services default value should be set to true. Other services' default value are optional.

packageName

Name assigned to the service package during deployment. The value you specify here must match the value specified during deployment.

Where used

Service

Required/Optional

Required if you use the SOAM_DEPLOY_DIR variable in the Service section; optional if you do not use that variable in the Service section.

Default

None

maxOtherInstances

Applies only when multiple services are used in a single application. Defines the maximum number of other service instances that a single service instance manager can run concurrently, in addition to the current service.

When this parameter is set to 0, no other service instances started by the service instance manager can run at the same time when this service instance manager starts this service. As a result, any other running service instances managed by this service instance manager are stopped when this service is started.

When this parameter is set to a number larger than 0, the service instance manager can keep maxOtherInstances service instances running along with this service concurrently. Setting a value larger than 0 saves service instance loading time, because service instances are not shut down and restarted.

When a service instance manager needs to shut down a service to satisfy maxOtherInstances, it first tries to gracefully shut down the service instance. If the service instance does not exit when the DestroyServiceTimeout (Service > Control > Method > Timeout, duration attribute) expires, the process is killed.

Note:

A service instance manager only manages multiple services that have the same serviceToSlotRatio. As soon as a service instance manager needs

to manage a service with a different serviceToSlotRatio, it will terminate all of the service instances it is managing and start a new service instance with the new ratio. The other service instances will not be restarted with the new serviceToSlotRatio unless they are required to run workload.

Where used

Service

Required/Optional

Optional

Default

0, only one service instance can run at a time.

Example

For example, if you have three services with different values for maxOtherInstances:

Name	maxOtherInstances	Default
S1	1	true
S2	1	false
S3	0	false

When the service instance manager starts, it starts service S1 by default. The service instance manager has one child process (S1). When service S2 is started, the service instance manager has 2 child processes (S1 and S2). If service S3 is started, the service instance manager has only one child process, which is S3 (S1 and S2 are shutdown by the service instance manager).

When service S1 is started again, the service instance manager has only one child process, which is S1 (S3 is shutdown by the service instance manager because it's maxOtherInstances is 0).

deploymentTimeout

Maximum time, in seconds, to wait before soamdeploy checks the connection with the repository server.

Where used

Service

Required/Optional

Optional

Default

300 seconds

Valid values

0—2147483647

deploymentRetryCount

Specifies the number of times soamdeploy retries to connect to the repository server and download the service package from the repository server. After the service experiences the specified number of retries, the service is aborted.

Where used

Service

Required/Optional

Optional

Default

3

Valid values

0 or Positive integer.

debugSetting

Helps you to use a debugger to step through the service code and find any errors in the service logic or in the environment.

For the customized debugSetting, you can configure the specific events that you want Symphony to detect by specifying the customizedDebugAction for each of these events.

Where used

Service

Required/Optional

Optional

Default

none

Valid values

none | customized | full

none—never generates debug logs

customized—generates debug logs upon detection of specific customizable events

full—always generate debug logs

cleanupTimeout

Maximum time, in seconds, to allow a service instance to cleanup if an application is disabled or unregistered, or a middleware component becomes unavailable.

Cleanup allows the running service method to gracefully finish executing. If the service method and cleanup does not complete within the set time, then Symphony will terminate the instance.

Where used

Service

Required/Optional

Optional

Default

60 seconds

Valid values

Positive integer

delaySlotRelease

Specifies the amount of time that an unassigned service will remain idle with the application before releasing its corresponding slot to EGO (and terminating the service). An unassigned service is defined as a service not assigned to any session. If new workload arrives before the configured time elapses, these unassigned services can be used to handle the new workload. Once an unassigned service becomes assigned to a session again, its idle timer is reset.

Where used

Service

Required/Optional

Optional

Default

Unspecified (use application-level setting)

Valid values

Non-negative integer

Service >osTypes >osType

Configuration related to the operating system in which services run.

name

Operating system environment in which to run the service. Symphony uses this information to determine where the executable for the service is located and set up the environment variables for the service instance.

Symphony supports:

- all—This configuration is applied if the osType for the allocated resource is not configured in the application profile.
- LINUX86—A 32-bit Linux-based environment, such as RedHat Linux
- NTX86—A 32-bit Windows-based environment, such as Windows 2000, Windows NT, and Windows XP
- X86_64—A 64-bit Linux-based environment
- NTX64—A 64-bit Windows-based environment

- SOL64—A Solaris SPARC based environment
- SOLX8664—A Solaris x86-64 based environment
- User-defined operating-system types, defined in the ego. shared file

Note:

While specifying PATH, ensure that you use the proper path delimiter (colon or semi-colon) according to the osType.

Where used

- Service > osTypes > osType

Required/Optional

Required

Valid values

all | LINUX86 | X86_64 | NTX86 | NTX64 | SOL64 | SOLX8664

logDirectory

Used for log retrieval through the Platform Management Console. Path to the directory in which logs are written by services for this application. The path can be any desired path and must be the same on all compute hosts.

Configure logDirectory, fileNamePattern, and subDirectoryPattern for each operating system type defined in your service section.

Note:

Log retrieval is only available in Symphony on the grid. Log retrieval is not available in Symphony DE.

Where used

Service > OsTypes > osType

Required/Optional

Optional

Default value

- Linux—\$SOAM_HOME/work
- Windows—%SOAM_HOME%\work

Related attributes

- fileNamePattern—File naming convention for service log files written to the logDirectory.
- subDirectoryPattern—Convention for naming subdirectories in the logDirectory.

fileNamePattern

Used for log retrieval through the Platform Management Console. File naming convention for service log files written to the logDirectory. Specify how your files are named. When retrieving logs for a specific task, the system looks for log files that partially match the file name pattern. If you are going to name files

with variables such as the ID or tag for sessions or tasks, specify the appropriate special variable; refer to the valid values for special variables in this section.

Configure `logDirectory`, `fileNamePattern`, and `subDirectoryPattern` for each operating system type defined in your service section.

Note:

Log retrieval is only available in Symphony on the grid. Log retrieval is not available in Symphony DE.

Where used

Service > OsTypes > osType

Required/Optional

Optional

Valid values

alphanumeric string

Special variables: `%taskId%`, `%sessionId%`, `%sessionTag%`, `%taskTag%`

Default value

empty, retrieve all files under `logDirectory/subDirectory`

Example

If log files have, for example, the service name, then sessionID, and task ID, for example, your log file name is `sampleService_123_1.log`, you can specify:

`sampleService_%sessionId%_%taskId%.log`

The system matches files containing the session ID and task ID.

Related attributes

- `logDirectory`—Path to the directory in which logs are written by services for this application.
- `subDirectoryPattern`—Convention for naming subdirectories in the `logDirectory`.

subDirectoryPattern

Used for log retrieval through the Platform Management Console. Convention for naming subdirectories in the `logDirectory`. If you want to name subdirectories according to variables such as the ID or tag for sessions or tasks, specify the appropriate special variable; refer to the valid values for special variables in this section. When retrieving logs, the system looks for log files in subdirectories that partially match the sub-directory pattern.

Configure `logDirectory`, `fileNamePattern`, and `subDirectoryPattern` for each operating system type defined in your service section.

Note:

Log retrieval is only available in Symphony on the grid. Log retrieval is not available in Symphony DE.

Where used

Service > OsTypes > osType

Required/Optional

Optional

Valid values

alphanumeric string

Special variables: %sessionId%, %taskId%, %sessionTag%, %taskTag%

Default value

empty, no subdirectories exist

Example

If you want to name subdirectories according to session ID, specify, without any backslashes:

%sessionId%

The system looks for logs in subdirectories named according to the session ID.

Related attributes

- logDirectory—Path to the directory in which logs are written by services for this application.
- fileNamePattern—File naming convention for service log files written to the logDirectory.

startCmd

Path to the program executable for the service. If you deploy a service package using Symphony deployment, the directory where your service package is extracted can be referred to as \$ {SOAM_DEPLOY_DIR}. If you need to reference a file in your service package, use the \$ {SOAM_DEPLOY_DIR} variable.

For example, on Windows, if in your service package you have the directory structure \myservice\myservice.exe, indicate \${SOAM_DEPLOY_DIR}\myservice\myservice.exe in startCmd.

For example, on Linux, if in your service package you have the directory structure /myservice/myservice, indicate \${SOAM_DEPLOY_DIR}/myservice/myservice in startCmd.

Note:

To run a Windows .bat script, you need to specify a special syntax to run the batch file in command shell. For example:

```
<osType name=NTX86" startCmd="cmd /c cmd /c install.bat"/>
```

Where used

Service > osTypes > osType

Required/Optional

Required

Valid values

Any location accessible on your network or local disk.

workDir

Working directory of the service instance.

Where used

- Service > osTypes > osType

Required/Optional

Optional

Valid values

Any location on local disk.

Default value

The default working directory for the service is:

- \${SOAM_HOME}/work

preExecCmd

When using a deployment tool other than Symphony soamdeploy, the command specified here will run on the compute host and can be used to copy the service program to that host.

Multiple commands can be run by calling a script or batch file containing the commands. The path to this command must be defined by an environment variable.

preExecCmd is called only when the packageName is not configured in the service section.

Where used

Service > osTypes > osType

Required/Optional

Optional

Default value

"" (empty)—No pre-execution command will be executed

Service > osTypes > osType > env

Configure environment variables for running service instances.

name

Name of an environment variable to set in the runtime environment of the service. You can also define environment variables so that they get substituted in the startCmd, preExecCmd, and workDir attributes.

One env statement is required for each environment variable.

No environment variables need to be set for the service.

You can refer to defined environment variables in other environment variables. For example:


```
<env name="ENVAR1">${SOAM_HOME}/work</env>
```

```
<env name="ENVAR2">${ENVAR1}/${VERSION_NUM}</env>
```

Symphony substitutes the following environment variables with system values:

- `${PATH}`—Specifies the path to the relevant executable. The specified path is pre-pended in front of the system Path at runtime. For Windows, specifies the path to any dependent library, including the Symphony libraries.
- `${LD_LIBRARY_PATH}`—Linux only. Specifies the path to the library where the GCC-specific Symphony files are located. The specified path is pre-pended in front of the system `LD_LIBRARY_PATH` at runtime
- `${SOAM_HOME}`—Directory where Symphony is installed. Replaced with the value of the operating system environment variable `SOAM_HOME` set on the host.
- `${VERSION_NUM}`—Symphony version on which the service is running.
- `${EGO_MACHINE_TYPE}`—Specifies the host type installation. For example, `win32-vc7` specifies a Windows machine. The variable is replaced with the value of the operating system environment variable `EGO_MACHINE_TYPE` set on the host.
- `${SOAM_DEPLOY_DIR}`—Internal system directory in which the service package is deployed.
- `${SOAM_SERVICE_EVENT_REPLAY_LOG}`—Service is driven by the events logged in the SERL file that this variable references. If this environment variable is not defined, service is driven by the SIM (as through Symphony).

Note:

`EGO_MACHINE_TYPE`, `SOAM_DEPLOY_DIR`, and `SOAM_SERVICE_EVENT_REPLAY_LOG` environment variables are reserved by Symphony, modification of these environment variables may result in undesirable behavior of your application.

Where used

- Service > osTypes > osType > env

Required/Optional

Required

<Service ><Control><Method> element name

Service API method to which defined events apply.

Where used

Service > Control > Method

Required/Optional

Required

Valid values

- Register
- CreateService

- SessionEnter
- SessionUpdate
- Invoke
- SessionLeave
- DestroyService

Default value

No default value

<Service ><Control><Method><Timeout> element

Defines how long to wait for the method specified in <Method name=...> element to complete, and what action to take on service instances, sessions, or tasks, upon timeout of the method.

duration

Number of seconds to wait for the method specified in <Method name=...> to complete before a timeout is considered.

Required/Optional

Required

Valid values

Method	Valid value
Register	1 or more
CreateService	0 or more
SessionEnter	0 or more
SessionUpdate	0 or more
Invoke	0 or more
SessionLeave	0 or more
DestroyService	0 or more

Note:

The method never times out if you set the value to 0.

Default values

Method	Default value
Register	60 seconds
CreateService	0 seconds, never timeout
SessionEnter	0 seconds, never timeout
SessionUpdate	0 seconds, never timeout
Invoke	0 seconds, never timeout

Method	Default value
SessionLeave	0 seconds, never timeout
DestroyService	15 seconds

actionOnSI

Action to take on the service instance when a timeout occurs.

Where used

Service > Control > Timeout

Required/Optional

Required

Valid values

- `blockHost`—When a timeout is reached on the method, terminate the running service instance on this host and do not use this host to start any other service instance for the application. The host on which the service instance was running is added to the blocked host list for the application. This host is no longer selected to run work for the application until it is explicitly unblocked through the EGO command line or the Platform Management Console.
- `restartService`—When a timeout is reached on the method, terminate the service instance, start a new service instance on the same host, and recover state. There is no limit to the number of times that a service instance can be restarted.

Not all values are possible with all methods. The table below outlines possible values:

Method	Possible action
Register	<ul style="list-style-type: none"> • <code>blockHost</code> • <code>restartService</code>
CreateService	<ul style="list-style-type: none"> • <code>blockHost</code> • <code>restartService</code>
SessionEnter	<ul style="list-style-type: none"> • <code>blockHost</code> • <code>restartService</code>
SessionUpdate	<ul style="list-style-type: none"> • <code>blockHost</code> • <code>restartService</code>
Invoke	<ul style="list-style-type: none"> • <code>blockHost</code> • <code>restartService</code>
SessionLeave	<ul style="list-style-type: none"> • <code>blockHost</code> • <code>restartService</code>
DestroyService	<ul style="list-style-type: none"> • No action possible on the service instance

Default values

Default values differ according to method. The following table outlines default values by method.

Method	Default action
Register	<ul style="list-style-type: none"> blockHost
CreateService	<ul style="list-style-type: none"> blockHost
SessionEnter	<ul style="list-style-type: none"> blockHost
SessionUpdate	<ul style="list-style-type: none"> blockHost
Invoke	<ul style="list-style-type: none"> restartService
SessionLeave	<ul style="list-style-type: none"> restartService
DestroyService	<ul style="list-style-type: none"> No action possible

actionOnWorkload

Action to take on sessions and tasks when a timeout occurs.

Where used

Service > Control > Method > Timeout

Required/Optional

Required

Valid values

- retry—When a timeout is reached on the method, retry the method up to the number of times configured by the session and task retry limits in the application profile.

For SessionEnter and SessionUpdate, the system attempts to bind the session to the service instance up to the sessionRetryLimit in the application profile before the session is aborted.

Note:

The retry count for both of these methods are considered together. For example, if SessionEnter fails once and SessionUpdate fails twice, then the session rerun count is equal to 3. Therefore the SessionRetryCount should be set to a value that accounts for both SessionEnter and SessionUpdate failures.

For Invoke, the system attempts to run the task up to the taskRetryLimit defined in the application profile before the task is failed.

- fail —When a timeout is reached on the method, abort the session or fail the task, and propagate errors to the client application.

For SessionEnter or SessionUpdate, immediately abort the session. Do not retry the method.

For Invoke, immediately fail the task. Do not retry the method.

Not all values are possible with all methods. The table below outlines possible actions for methods:

Method	Possible action
Register	<ul style="list-style-type: none"> No action possible
CreateService	<ul style="list-style-type: none"> No action possible
SessionEnter	<ul style="list-style-type: none"> retry fail
SessionUpdate	<ul style="list-style-type: none"> retry fail
Invoke	<ul style="list-style-type: none"> retry fail
SessionLeave	<ul style="list-style-type: none"> No action possible
DestroyService	<ul style="list-style-type: none"> No action possible

Default values

Default values differ according to method. The following table outlines default values by method for actions taken.

Method	Default action
Register	<ul style="list-style-type: none"> No action possible
CreateService	<ul style="list-style-type: none"> No action possible
SessionEnter	<ul style="list-style-type: none"> retry
SessionUpdate	<ul style="list-style-type: none"> retry
Invoke	<ul style="list-style-type: none"> retry
SessionLeave	<ul style="list-style-type: none"> No action possible
DestroyService	<ul style="list-style-type: none"> No action possible

customizedDebugAction

Action to take on sessions and tasks when a timeout occurs and service debugsetion="customized".

Where used

Service > Control > Method > Timeout

Required/Optional

Optional

Valid values

- `writeServiceEventReplayFiles`—When Symphony detects that the specified method has timed out, it generates service event replay files to capture the relevant events that lead up to the timeout. This is the recommended setting if method timeout is an unexpected event for your service.
- `none`—When Symphony detects that the specified method has timed out, it does not generate service event replay files. This is the recommended setting if your service times out as a normal occurrence.

Default value

`writeServiceEventReplayFiles`

<Service ><Control><Method><Exit> element

Defines what action to take on service instances, sessions, or tasks when the method specified with the <Method> element exits.

actionOnSI

Action to take on the service instance when the method exits.

Where used

Service > Control > Method > Exit

Required/Optional

Required

Valid values

- `blockHost`—When the method exits, terminate the running service instance on this host and do not use this host to start any other service instance for the application. The host on which the service instance was running is added to the blocked host list for the application. This host is no longer selected to run work for the application until it is explicitly unblocked through the EGO command line or the Platform Management Console.
- `restartService`—When the method exits, terminate the service instance, start a new service instance on the same host, and recover state. There is no limit to the number of times that a service instance can be restarted.

Note:

For `DestroyService`, there is no action is possible on the service instance.

Default values

Default values differ according to method. The following table outlines default values by method for actions.

Method	Default action
Register	<ul style="list-style-type: none"> • <code>blockHost</code>

Method	Default action
CreateService	<ul style="list-style-type: none"> • blockHost
SessionEnter	<ul style="list-style-type: none"> • blockHost
SessionUpdate	<ul style="list-style-type: none"> • blockHost
Invoke	<ul style="list-style-type: none"> • restartService
SessionLeave	<ul style="list-style-type: none"> • restartService
DestroyService	<ul style="list-style-type: none"> • No action possible

actionOnWorkload

Action to take on sessions and tasks when the method exits.

Where used

Service > Control > Method > Exit

Required/Optional

Required

Valid values

- **retry**—If the service exits during execution of the specified method, retry the method up to the number of times configured by the session and task retry limits in the application profile.

For SessionEnter and SessionUpdate, the system attempts to bind the session to the service instance up to the sessionRetryLimit in the application profile before the session is aborted.

Note:

The retry count for both of these methods are considered together. For example, if SessionEnter fails once and SessionUpdate fails twice, then the session rerun count is equal to 3. Therefore the SessionRetryCount should be set to a value that accounts for both SessionEnter and SessionUpdate failures.

For Invoke, the system attempts to run the task up to the taskRetryLimit defined in the application profile before the task is failed.

- **fail** —When the method exits, abort the session or fail the task, and propagate errors to the client application.

For SessionEnter or SessionUpdate, immediately abort the session. Do not retry the method.

For Invoke, immediately fail the task. Do not retry the method.

Not all values are possible with all methods. The table below outlines possible actions for methods:

Method	Possible action
Register	<ul style="list-style-type: none"> No action possible
CreateService	<ul style="list-style-type: none"> No action possible
SessionEnter	<ul style="list-style-type: none"> retry fail
SessionUpdate	<ul style="list-style-type: none"> retry fail
Invoke	<ul style="list-style-type: none"> retry fail
SessionLeave	<ul style="list-style-type: none"> No action possible
DestroyService	<ul style="list-style-type: none"> No action possible

Default values

Default values differ according to method. The following table outlines default values by method for actions taken.

Method	Default action
Register	<ul style="list-style-type: none"> No action possible
CreateService	<ul style="list-style-type: none"> No action possible
SessionEnter	<ul style="list-style-type: none"> retry
SessionUpdate	<ul style="list-style-type: none"> retry
Invoke	<ul style="list-style-type: none"> retry
SessionLeave	<ul style="list-style-type: none"> No action possible
DestroyService	<ul style="list-style-type: none"> No action possible

customizedDebugAction

Action to take on sessions and tasks when the service process exits during the execution of the specified method and service debugSetting="customized"..

Where used

Service > Control > Method > Exit

Required/Optional

Optional

Valid values

- `writeServiceEventReplayFiles`—When Symphony detects that the service process has exited (or crashed) while executing the specified method, it generates service event replay files to capture the relevant events that lead up to the exit. This is the recommended setting if the service process exiting or crashing in the specified method is an unexpected event for your service.
- `none`—When Symphony detects that the service process has exited (or crashed) during execution of the specified method, it does not generate service event replay files. This is the recommended setting if your service exits as a normal occurrence.

Default value

`writeServiceEventReplayFiles`

<Service><Control><Method><Return> element

Defines what action to take on service instances, sessions, or tasks when the method specified in <Method name=...> returns normally (controlCode=0), or if a control code is defined, what action to take when the method returns with the specified control code.

actionOnSI

Action to take on the service instance when the method returns normally or with the specified control code.

Where used

Service > Control > Method > Return

Required/Optional

Required

Valid values

Note:

Actions on return cannot be configured for Register or DestroyService

- `blockHost`—When the method returns normally or with a specified control code, terminate the running service instance on this host and do not use this host to start any other service instance for the application.

The host on which the service instance was running is added to the blocked host list for the application. This host is no longer selected to run work for the application until it is explicitly unblocked through the EGO command line or the Platform Management Console.
- `restartService`—When the method returns normally or with a specified control code, terminate the service instance, start a new service instance on the same host, and recover state. There is no limit to the number of times that a service instance can be restarted.
- `keepAlive`—When the method returns normally or with a specified control code, take no action on the running service instance. Return an error only.

Default values

Method	Default value
Register	<ul style="list-style-type: none"> No action possible
CreateService	<ul style="list-style-type: none"> keepAl i ve
SessionEnter	<ul style="list-style-type: none"> keepAl i ve
SessionUpdate	<ul style="list-style-type: none"> keepAl i ve
Invoke	<ul style="list-style-type: none"> keepAl i ve
SessionLeave	<ul style="list-style-type: none"> keepAl i ve
DestroyService	<ul style="list-style-type: none"> No action possible

controlCode

Numeric identifier that indicates to the system what action to take based on the number that is returned by the method specified in the <Method> element. There can be multiple control codes defined per method, and different actions to take based on control code.

Where used

Service > Control > Method > Return

Required/Optional

Required

Valid values

integers, positive and negative

Default value

0, default action taken on workload and service instances, as specified by default actions on actionOnSI and actionOnWorkload.

actionOnWorkload

Action to take on sessions and tasks when the method returns normally, or with a specified control code.

Where used

Service > Control > Method > Return

Required/Optional

Required

Valid values

- **retry**—When the method returns normally or with the specified control code, retry the method up to the number of times configured by the session and task retry limits in the application profile.

For SessionEnter and SessionUpdate, the system attempts to bind the session to the service instance up to the sessionRetryLimit in the application profile before the session is aborted.

Note:

The retry count for both of these methods are considered together. For example, if SessionEnter fails once and SessionUpdate fails twice, then the session rerun count is equal to 3. Therefore the SessionRetryCount should be set to a value that accounts for both SessionEnter and SessionUpdate failures.

For Invoke, the system attempts to run the task up to the taskRetryLimit defined in the application profile before the task is failed.

- **fail** —When the method returns normally or with the specified control code, abort the session or fail the task, and propagate errors to the client application.

For SessionEnter or SessionUpdate, immediately abort the session. Do not retry the method.

For Invoke, immediately fail the task. Do not retry the method.

- **succeed**—This is a normal return. This is treated as a successful execution of SessionEnter, SessionUpdate, or Invoke.

For Invoke, the task completes successfully (in the Done state).

Not all values are possible with all methods. The table below outlines possible actions for methods:

Method	Possible value
Register	<ul style="list-style-type: none"> • No action possible
CreateService	<ul style="list-style-type: none"> • No action possible
SessionEnter	<ul style="list-style-type: none"> • retry • fail • succeed
SessionUpdate	<ul style="list-style-type: none"> • retry • fail • succeed
Invoke	<ul style="list-style-type: none"> • retry • fail • succeed
SessionLeave	<ul style="list-style-type: none"> • No action possible
DestroyService	<ul style="list-style-type: none"> • No action possible

Default values

Default values differ according to method.

Method	Default value
Register	<ul style="list-style-type: none"> No action possible
CreateService	<ul style="list-style-type: none"> No action possible
SessionEnter	<ul style="list-style-type: none"> succeed
SessionUpdate	<ul style="list-style-type: none"> succeed
Invoke	<ul style="list-style-type: none"> succeed
SessionLeave	<ul style="list-style-type: none"> No action possible
DestroyService	<ul style="list-style-type: none"> No action possible

customizedDebugAction

Action to take on the service instance when the method returns normally or with the specified control code when service debugSetting="customized".

Where used

Service > Control > Method > Return

Required/Optional

Optional

Valid values

- writeServiceEventReplayFiles—When Symphony detects that the specified method has returned normally with or without a specific control code, it generates service event replay files.
- none—When Symphony detects that the specified method has returned normally with or without a specific control code, it does not generate service event replay files. This is the recommended setting, as this is typically a normal, successful event.

Default value

none

<Service ><Control><Method><Exception> element

Defines what action to take on service instances, sessions, or tasks when the method specified in <Method name=...> encounters a fatal or failure exception.

type

Exception type to which the configured action applies. Exception types cannot be configured for Register or DestroyService.

Where used

Service > Control > Method > Exception

Required/Optional

Required

Valid values

- `failure`— Exception type is `FailureException`.
- `fatal` — Exception type is `FatalException`.

Default values

No default value

controlCode

Numeric identifier that indicates to the system what action to take based on the number that is returned by the specified exception. There can be multiple control codes defined per exception, and different actions to take based on control code.

Where used

Service > Control > Method > Exception

Required/Optional

Required

Valid values

integers, positive and negative

Default value

0, default action taken on workload and service instances, as specified by default actions on `actionOnSI` and `actionOnWorkload`.

actionOnSI

Action to take on the service instance when a failure or fatal exception occurs.

Where used

Service > Control > Method > Exception

Required/Optional

Required

Valid values

- `blockHost`—When the specified exception occurs, terminate the running service instance on this host and do not use this host to start any other service instance for the application.

The host on which the service instance was running is added to the blocked host list for the application. This host is no longer selected to run work for the application until it is explicitly unblocked through the EGO command line or the Platform Management Console.

- **restartService**—When the specified exception occurs, terminate the service instance, start a new service instance on the same host, and recover state. There is no limit to the number of times that a service instance can be restarted.
- **keepAlive**—When the specified exception occurs, take no action on the running service instance.

Note:

Actions cannot be configured for Register or DestroyService

Method	Possible values Fatal Exception	Possible values Failure Exception
Register	No action possible	No action possible
CreateService	<ul style="list-style-type: none"> • blockHost • restartService 	<ul style="list-style-type: none"> • blockHost • restartService
SessionEnter	<ul style="list-style-type: none"> • blockHost • restartService • keepAlive 	<ul style="list-style-type: none"> • blockHost • restartService • keepAlive
SessionUpdate	<ul style="list-style-type: none"> • blockHost • restartService • keepAlive 	<ul style="list-style-type: none"> • blockHost • restartService • keepAlive
Invoke	<ul style="list-style-type: none"> • blockHost • restartService • keepAlive 	<ul style="list-style-type: none"> • blockHost • restartService • keepAlive
SessionLeave	<ul style="list-style-type: none"> • blockHost • restartService • keepAlive 	<ul style="list-style-type: none"> • blockHost • restartService • keepAlive
DestroyService	No action possible	No action possible

Default values

Method	Default values Fatal Exception	Default values Failure Exception
Register	No action possible	No action possible
CreateService	<ul style="list-style-type: none"> • blockHost 	<ul style="list-style-type: none"> • blockHost
SessionEnter	<ul style="list-style-type: none"> • keepAlive 	<ul style="list-style-type: none"> • keepAlive
SessionUpdate	<ul style="list-style-type: none"> • keepAlive 	<ul style="list-style-type: none"> • keepAlive

Method	Default values Fatal Exception	Default values Failure Exception
Invoke	<ul style="list-style-type: none"> keepAlive 	<ul style="list-style-type: none"> keepAlive
SessionLeave	<ul style="list-style-type: none"> keepAlive 	<ul style="list-style-type: none"> keepAlive
DestroyService	No action possible	No action possible

actionOnWorkload

Action to take on sessions and tasks when a fatal or failure exception occurs.

Where used

Service > Control > Method > Exception

Required/Optional

Required

Valid values

- retry—When the specified exception occurs, retry the method up to the number of times configured by the session and task retry limits in the application profile.

For SessionEnter and SessionUpdate, the system attempts to bind the session to the service instance up to the sessionRetryLimit in the application profile before the session is aborted.

Note:

The retry count for both of these methods are considered together. For example, if SessionEnter fails once and SessionUpdate fails twice, then the session rerun count is equal to 3. Therefore the SessionRetryCount should be set to a value that accounts for both SessionEnter and SessionUpdate failures.

For Invoke, the system attempts to run the task up to the taskRetryLimit defined in the application profile before the task is failed.

- fail —When the specified exception occurs, abort the session or fail the task, and propagate errors to the client application.

For SessionEnter or SessionUpdate, immediately abort the session. Do not retry the method.

For Invoke, immediately fail the task. Do not retry the method.

Not all values are possible with all methods. The table below outlines possible actions for methods:

Method	Possible values Fatal Exception	Possible values Failure Exception
Register	No action possible	No action possible
CreateService	No action possible	No action possible

Method	Possible values Fatal Exception	Possible values Failure Exception
SessionEnter	<ul style="list-style-type: none"> • retry • fail 	<ul style="list-style-type: none"> • retry • fail
SessionUpdate	<ul style="list-style-type: none"> • retry • fail 	<ul style="list-style-type: none"> • retry • fail
Invoke	<ul style="list-style-type: none"> • retry • fail 	<ul style="list-style-type: none"> • retry • fail
SessionLeave	No action possible	No action possible
DestroyService	No action possible	No action possible

Default values

Default values differ according to method.

Method	Default values Fatal Exception	Default values Failure Exception
Register	No action possible	No action possible
CreateService	No action possible	No action possible
SessionEnter	<ul style="list-style-type: none"> • fail 	<ul style="list-style-type: none"> • retry
SessionUpdate	<ul style="list-style-type: none"> • fail 	<ul style="list-style-type: none"> • retry
Invoke	<ul style="list-style-type: none"> • fail 	<ul style="list-style-type: none"> • retry
SessionLeave	No action possible	No action possible
DestroyService	No action possible	No action possible

customizedDebugAction

Action to take on sessions and tasks when a fatal or failure exception occurs in the specified method and debugSetting="customized".

Where used

Service > Control > Method > Exception

Required/Optional

Optional

Valid values

- writeServiceEventReplayFiles—When Symphony detects that the specified method has thrown a particular exception (FatalException or FailureException), as specified, it generates service event replay files to capture the relevant events that lead up to the exception.

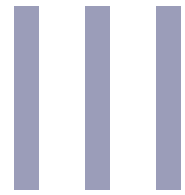
Service section

- none—When Symphony detects that the specified method has thrown a particular exception (FatalException or FailureException), as specified, it does not generate service event replay files.

Default value

writeServiceEventReplayFiles

P A R T



Daemons

egosc

EGO service controller.

Synopsis

egosc [-d *conf_dir*]

egosc [-C *esc_conf_dir*]

egosc -h

egosc -V

Description

The service controller is the first service that runs on top of the EGO kernel. It functions as a bootstrap mechanism for starting the other services in the cluster. It also monitors and recovers the other services. It is somewhat analogous to `init` on UNIX systems or Service Control Manager on Windows systems. After the kernel boots, it reads a configuration file to retrieve the list of services to be started.

The service controller acts as a client to the EGO kernel, requesting resource allocations for running services and instantiating activities to host those services. It ensures that all the services are running by detecting failures and restarting service instances based on the parameter settings in the Control Policy portion of the service profile.

`egosc` is started automatically by `vemkd` and exits when `vemkd` exits. Under normal circumstances, you do not need to start it manually.

Caution:

Never start the daemon manually without checking the options: specify the `-V` option to check the version, the `-d` option to start the daemon in debug mode, or the `-C` option to validate its configuration files.

Options

-d *conf_dir*

Starts the daemon, reading from the EGO configuration file `ego.conf` in the specified directory, rather than from the directory set via the `EGO_CONFDIR` environment variable.

Use this option when starting the daemon in debug mode.

Caution:

Never start the daemon manually unless directed to do so by Platform Technical Support.

-C *esc_conf_dir*

Starts the daemon to validate its configuration files in the specified directory and then exits.

Caution:

Never start the daemon manually unless directed to do so by Platform Technical Support.

-h

Outputs command usage and exits.

-v

Outputs product version and exits.

Files

ego.conf

egosc reads the configuration file `ego.conf` to retrieve the location of the service controller configuration information, specified by the `EGO_ESRVDIR` parameter.

egosc_conf.xml

egosc reads the configuration file `egosc_conf.xml` to retrieve the following information:

ESC_PORT

The TCP port egosc uses to serve requests.

ESC_LOGDIR

The directory where egosc logs error or debug messages.

ESC_LOG_MASK

The log level used to determine the amount of detail logged.

ESC_AUDIT_LOG

Whether audit logging is turned on or off.

ESC_TS_PARAMS

SSL configuration parameters for the Service Controller.

services files

egosc looks in the directory `EGO_ESRVDIR/esc/conf/services` to locate the services that it is to manage. Each service has a corresponding file in the `services` directory.

lim

Load information daemon or service, monitoring host load.

Synopsis

```
lim [-d conf_dir] [-c license_file] [-debug_level]
```

```
lim -C
```

```
lim -t
```

```
lim -h
```

```
lim -V
```

Description

There is one lim daemon/service on every host in the cluster. Of these, one lim from the master list is elected master lim for the cluster. The master lim receives load information from the other lim daemons, and provides services to all host.

The lim does the following for the host on which it runs:

- Starts pem on that host
- Provides system configuration information to vemkd
- Monitors load and provides load information statistics to vemkd and users

The master lim starts vemkd and pem on the master host.

The non-master lim daemons monitor the status of the master `lim` and elect a new master (from the master list) if the current master `lim` becomes unavailable.

Collectively, the lims in the cluster coordinate the collection and transmission of load information. Load information is collected in the form of load indices.

Caution:

Never start the daemon manually without options: specify the `-V` option to check the version, the `-d` option to start the daemon in debug mode, or the `-C` option to validate its configuration files.

Options

-d *conf_dir*

Starts the daemon, reading from the EGO configuration file `ego.conf` in the specified directory, rather than from the directory set via the `EGO_CONFDIR` environment variable.

Use this option when starting the daemon in debug mode.

Caution:

Never start the daemon manually unless directed to do so by Platform Technical Support.

-c *license_file*

Specifies an alternative location to look for the license file rather than the default directory or the directory specified by the EGO_LICENSE_FILE environment parameter in `ego.conf`.

Specify the full path to the license file including the file name.

Caution:

Never start the daemon manually unless directed to do so by Platform Technical Support.

-debug_level

Starts the lim in debug mode. When running in debug mode, the lim uses a hard-coded port number rather than the one registered in system services.

Specify one of the following values:

-1

Starts the lim in the background, with no associated control terminal.

-2

Starts the lim in the foreground, displaying the log messages to the terminal.

Caution:

Never start the daemon manually unless directed to do so by Platform Technical Support.

-t

Displays host information, such as host type, host architecture, number of physical processors, number of cores per physical processor, number of threads per core, and license requirements.

Note:

When running Linux kernel version 2.4, you must run `lim -t` as root to ensure consistent output with other clustered application management commands (for example, output from running Platform LSF command `lshosts`).

-h

Outputs command usage and exits.

-v

Outputs product version and exits.

Files

ego.conf

The lim reads the configuration file `ego.conf` to retrieve configuration information. `ego.conf` is a generic configuration file shared by all daemons/services and clients. It

contains configuration information and other information that dictates the behavior of the software.

Some of the parameters lim retrieves from `ego.conf` are as follows:

EGO_LIM_PORT

The TCP port the lim uses to serve all applications.

EGO_SERVERDIR

The directory used for reconfiguring the lim—where the lim binary is stored.

EGO_LOGDIR

The directory used for message logs.

EGO_LOG_MASK

The log level used to determine the amount of detail logged.

EGO_DEBUG_LIM

The log class setting for the `lim`.

EGO_LICENSE_FILE

The full path to and name of the EGO license file.

EGO_DEFINE_NCPUS

Defines whether `ncpus` is to be defined as `procs`, `cores`, or `threads`. This parameter overrides `LSF_ENABLE_DUALCORE`. If `EGO_ENABLE_DUALCORE` is set, `EGO_DEFINE_NCPUS` settings take precedent.

- `procs` (if `ncpus` defined as `procs`, then `ncpus` = `nprocs`)
- `cores` (if `ncpus` defined as `cores`, then `ncpus` = `nprocs` x `ncores`)
- `threads` (if `ncpus` defined as `threads`, then `ncpus` = `nprocs` x `ncores` x `nthreads`)

Note:

When `EGO_DEFINE_NCPUS` is set, run queue-length values (`r1*` values returned by `lsload`) are automatically normalized based on the set value.

If `EGO_DEFINE_NCPUS` is not defined, but `EGO_ENABLE_DUALCORE` is set, the lim reports the number of cores. If both `EGO_DEFINE_NCPUS` and `LSF_ENABLE_DUALCORE` are set, then the `EGO` parameter takes precedence.

EGO_ENABLE_DUALCORE

Defines if the hosts have dual cores or not. Is overridden by `EGO_DEFINE_NCPUS`, if set.

Note:

lim

If EGO_DEFINE_NCPUS is not defined, but EGO_ENABLE_DUALCORE is set, the lim reports the number of cores. If both EGO_DEFINE_NCPUS and LSF_ENABLE_DUALCORE are set, then the EGO parameter takes precedence.

Customization

You can customize the `lim` by changing configuration files in EGO_CONFDIR directory. Configure `ego.cluster.<cluster_name>` to define various cluster properties such as the resources on individual hosts, the load threshold values for a host, and so on. Configure `ego.shared` to define host models read by the `lim`, or the CPU factor of individual hosts.

pem

Process execution daemon, monitoring execution.

Synopsis

pem [-d *conf_dir*] [-1 | -2]

pem -h

pem -V

Description

There is at least one pem daemon on every host in the cluster: on Windows, there is one per host; on Linux, there is one pem daemon per host, plus one for every activity running on that host—the number varies with the number of activities running at any given time.

The pem daemon starts all activities and monitors the activity life cycle.

The pem daemon is started automatically by the `l i m` daemon, and exits when `l i m` exits.

Caution:

Never start pem manually except with the `-V` option to check the version configuration.

Options

-d *conf_dir*

Starts the daemon, reading from the EGO configuration file `ego.conf` in the specified directory, rather than from the directory set via the `EGO_CONFDIR` environment variable.

Use this option when starting the daemon in debug mode.

Caution:

Never start the daemon manually unless directed to do so by Platform Technical Support.

-1

Starts the daemon in the background, with no associated control terminal. Outputs log messages into log file.

-2

Starts the daemon in debug mode. When running in debug mode, the daemon runs in the foreground, displaying the log messages. Outputs log messages to stderr.

Caution:

Never start the daemon manually unless directed to do so by Platform Technical Support.

-h

pem

Outputs command usage and exits.

-v

Outputs product version and exits.

Files

ego.conf

pem reads the configuration file `ego.conf` to retrieve the location of its configuration information. `ego.conf` is a generic configuration file shared by all daemons and clients. It contains configuration information and other information that dictates the behavior of the software.

pem reads `ego.conf` to retrieve the following information:

EGO_PEM_PORT

The TCP port pem uses to serve all requests.

EGO_LOGDIR

The directory used for message logs.

EGO_TMPDIR

The temporary directory pem uses for persistent files. If not specified, uses your location set in the `$TMPDIR` or `%TMPDIR%` environment variable. If that is not set either, defaults to your operating system's default temporary directory (for example, on Linux: `/tmp`).

If `EGO_LOGDIR` is not defined in `ego.conf` but `EGO_TMPDIR` is defined, then logs for pem and vemkd are logged here.

EGO_LOG_MASK

The log level used to determine the amount of detail logged.

EGO_DEBUG_PEM

The log class setting for pem.

pim

Process information daemon, collecting resource usage of running processes.

Synopsis

pim [-h] [-V] [-d *conf_dir*] [-debug_level]

Description

The load information manager (*lim*) starts the *pim* daemon on every host participating in load sharing. The *pim* collects resource usage of the processes running on the local host. The information collected by the *pim* is used by clustered application managers (such as Platform LSF) to monitor resource consumption and enforce usage limits.

The *pim* updates the process information every 15 minutes unless a clustered application manager queries this information. If queried, the *pim* updates the process information every `EGO_PIM_SLEEP_TIME` seconds (which is defined in the `ego.conf` file). If not defined, the default value for `EGO_PIM_SLEEP_TIME` is 15 seconds. If the information is not queried for more than 5 minutes, the *pim* reverts back to the 15 minute update period.

The process information is stored in `EGO_PIM_INFODIR/pim.info.<hostname>` where `EGO_PIM_INFODIR` is defined in the `ego.conf` file. If this parameter is not defined, the default directory is `/tmp`. The *pim* daemon also reads this file when it starts up so that it can accumulate the resource usage of dead processes for existing process groups.

Note:

The *pim* daemon needs read access to `/dev/kmem` or its equivalent.

Options

-h

Outputs command usage and exits.

-V

Outputs product version and exits.

-d *env_dir*

Read `ego.conf` from the directory `env_dir`, rather than the default directory `/etc`, or the directory specified by the `SEGO_CONFDIR` environment variable.

-debug_level

Starts the *pim* in debug mode. When running in debug mode, the *pim* uses a hard-coded port number rather than the one registered in system services. The `debug_level` option overrides the parameter `EGO_LIM_DEBUG` defined in `ego.conf`.

Specify one of the following values:

-1

Starts the *lim* in the background, with no associated control terminal.

-2

pim

Starts the lim in the foreground, displaying the log messages to the terminal.

Files

/etc/ego.conf (by default) or EGO_CONFDIR/ego.conf

vemkd

EGO kernel daemon.

Synopsis

vemkd [-d *conf_dir*] [-2]

vemkd -C

vemkd -h

vemkd -V

Description

There is only one vemkd daemon in a cluster. It runs on the master host.

The vemkd daemon does the following:

- Starts the service controller daemon egosc
- Maintains security policies, allowing only authorized access
- Maintains resource allocation policies, distributing resources accordingly
- Serves as an information center where clients can query information about the cluster

The vemkd daemon is started automatically by the l i m daemon, and exits when l i m exits.

Caution:

Never start vemkd manually except with the - V option to check the version configuration.

Options

-d conf_dir

Starts the daemon, reading from the EGO configuration file *ego.conf* in the specified directory, rather than from the directory set via the EGO_CONFDIR environment variable.

Use this option when starting the daemon in debug mode.

Caution:

Never start the daemon manually unless directed to do so by Platform Technical Support.

-2

Starts the daemon in debug mode. When running in debug mode, the daemon runs in the foreground, displaying the log messages.

Caution:

Never start the daemon manually unless directed to do so by Platform Technical Support.

-C

vemkd

Checks the configuration of the daemon and then exits.

-h

Outputs command usage and exits.

-V

Outputs product version and exits.

EGO client

You can run an EGO client from any host to interact with vemkd, provided the client host has the environment set correctly with the EGO_CONFDIR environment variable pointing to the correct ego.conf file.

When the client has established a connection with vemkd, it can query general information about the cluster. To perform any control operations, the client needs to be logged on as an EGO user. You need to log on with at least consumer user access to request a resource from EGO, and with cluster administrator privileges to perform cluster-wide maintenance.

Files

ego.conf

vemkd reads the configuration file `ego.conf` to retrieve the location of its configuration information. `ego.conf` is a generic configuration file shared by all daemons and clients. It contains configuration information and other information that dictates the behavior of the software.

vemkd reads `ego.conf` to retrieve the following information:

EGO_KD_PORT

The TCP port vemkd uses to serve all requests.

EGO_WORKDIR

The directory used for data persistence.

EGO_LOGDIR

The directory used for message logs.

EGO_LOG_MASK

The log level used to determine the amount of detail logged.

EGO_DEBUG_KD

The log class setting for vemkd.

users.xml

Defines the EGO user accounts for the cluster.

P A R T

IV

Symphony Resources

Host properties

For a detailed list of host properties, refer to the Platform Management Console (PMC) online help.

Load indices

Index	Measures	Units	Direction	Averaged over	Update Interval
st at us	host status	string			15 seconds
r 15s	run queue length	processes	increasing	15 seconds	15 seconds
r 1m	run queue length	processes	increasing	1 minute	15 seconds
r 15m	run queue length	processes	increasing	15 minutes	15 seconds
ut	CPU utilization	percent	increasing	1 minute	15 seconds
pg	paging activity	pages in + pages out per second	increasing	1 minute	15 seconds
l s	logins	users	increasing	N/A	30 seconds
i t	idle time	minutes	decreasing	N/A	30 seconds
swp	available swap space	MB	decreasing	N/A	15 seconds
mem	available memory	MB	decreasing	N/A	15 seconds
t mp	available space in temporary file system	MB	decreasing	N/A	120 seconds
i o	disk I/O	KB per second	increasing	1 minute	15 seconds
freesl o t	available CPU slots	CPU slots			60 seconds

CPU run queue lengths (r15s, r1m, r15m)

The r 15s, r 1m and r 15m load indices are the 15-second, 1-minute and 15-minute average CPU run queue lengths. This is the average number of processes ready to use the CPU during the given interval.

On Linux/UNIX, run queue length indices are not necessarily the same as the load averages printed by the `upt i me (1)` command; `upt i me` load averages on some platforms also include processes that are in short-term wait states (such as paging or disk I/O).

Effective run queue length

On multiprocessor systems, more than one process can execute at a time. The run queue value on multiprocessor systems is scaled to make the CPU load of uniprocessors and multiprocessors comparable. The scaled value is called the effective run queue length.

Normalized run queue length

The CPU run queue length is adjusted based on the relative speeds of the processors (the CPU factor). The normalized run queue length is adjusted for both number of processors and CPU speed. The host with the lowest normalized run queue length runs a CPU-intensive job the fastest.

CPU utilization (ut)

The `ut` index measures CPU utilization, which is the percentage of time spent running system and user code. A host with no process running has a `ut` value of 0 percent; a host on which the CPU is completely loaded has a `ut` of 100 percent.

Paging rate (pg)

The `pg` index gives the virtual memory paging rate in pages per second. This index is closely tied to the amount of available RAM memory and the total size of the processes running on a host; if there is not enough RAM to satisfy all processes, the paging rate is high. Paging rate is a good measure of how a machine responds to interactive use; a machine that is paging heavily feels very slow.

Login sessions (ls)

The `ls` index gives the number of users logged in. Each user is counted once, no matter how many times they have logged into the host.

Interactive idle time (it)

On Linux/UNIX, the `it` index is the interactive idle time of the host, in minutes. Idle time is measured from the last input or output on a directly attached terminal or a network pseudo-terminal supporting a login session. This does not include activity directly through the X server such as CAD applications or emacs windows, except on Solaris and HP-UX systems.

On Windows, the `it` index is based on the time a screen saver has been active on a particular host.

Temporary directories (tmp)

The `tmp` index is the space available in MB on the file system that contains the temporary directory:

- `/tmp` on Linux/UNIX
- `C: \temp` on Windows

Swap space (swp)

The `swp` index gives the currently available virtual memory (swap space) in MB. This represents the largest process that can be started on the host.

Memory (mem)

The `mem` index is an estimate of the real memory currently available to user processes. This represents the approximate size of the largest process that could be started on a host without causing the host to start paging.

Free memory is calculated as a sum of physical free memory, cached memory, buffered memory and an adjustment value.

I/O rate (io)

The `io` index measures I/O throughput to disks attached directly to this host, in KB per second. It does not include I/O to disks that are mounted from other hosts.

Free slots (freeslot)

Current number of available slots for this host across ALL resource groups.

-R res_req

-R res_req

Selects the most appropriate resource for a particular purpose.

Synopsis

-R "select(*select_string*)"

-R "select(*select_string*) order(*order_string*)"

Description

A resource requirement string describes the criteria for defining a set of resources.

The entire resource requirement string cannot contain more than 512 characters.

If the characters "-" or "." form a part of the host or resource name, enclose the name using single quotation marks (for example, when a full host name is used, such as 'gr4e01.domain.name.com').

Examples of proper quotation mark usage include the following:

```
egosh resource list -R "select('host1.domain.name.com')"
```

```
egosh resource list -R "select('host1-1')"
```

```
egosh resource list -R "select('host1-1' || 'host1-2')"
```

Important:

If the command is issued in whole from the shell console or the requirement has white space, enclose the requirement in double quotation marks. For example:

```
egosh resource list -R "select(mem>100)"
```

If the command is issued from the egosh console, quotation marks are optional. For example:

```
egosh> resource list -R select(mem>100)
```

Options

select(*select_string*)

Specifies the criteria for selecting the resources. The selection string filters out the resources that do not meet the criteria, resulting in a list of one or more eligible resources.

The parentheses must be typed as shown.

order(*order_string*)

Specifies the sort order for selecting the best resource from the list of eligible resources. The most appropriate resource is listed first, the least is listed last.

The parentheses must be typed as shown.

select

The selection string excludes unsuitable resources by specifying the characteristics a resource must have to match a resource requirement.

select synopsis

select(*expression*)**select**(*expression operator expression*)**select**((*expression operator expression*) *operator expression*)

select description

The selection string is a logical expression used to select one or more resources to match one or more criteria. Any resource that satisfies the criteria is selected.

The selection string is used in many ways to select resources.

- To define resource groups in EGO
- By Platform Symphony application profiles to define the resources to run SOA workload
- To define resource requirements for both jobs and queues in Platform LSF

The resource selection string uses values for host_name, model, type, and/or resources as selection string expressions. These can be seen in the output of

```
egosh resource view
```

. When entering a resource requirement string in EGO, omit the operator and use only the value. For example:

Incorrect syntax: `select (type==linux86)`

Correct syntax: `select (linux86)`

Other examples of correct syntax:

```
select (linux86 && maxmem > 500)
select (maxmem > 2046 && LINUX86) => ib06b09
select (maxmem > 2046 && mg) => ib06b09
select (NTX86) => host1
```

When you input a string, ensure that you use valid characters. This requirements applies to all resource requirement strings. Valid characters include the following: `a-z A-Z 0-9 * / ! () . | \ ^ & $ # @ ~ % ``

select expression synopsis

resource_name operator value

resource_name

Specifies the name of the resource to use as selection criteria.

You can specify a static resource or a load index, depending on the purpose of the selection string.

operator

The following operators can be used in selection strings. If you are using the selection string in an XML format, you must use the applicable escape characters in the XML Equivalence column. The operators are listed in order of decreasing precedence:

Operator	XML Equivalent	Syntax	Meaning
!	n/a	!a	Logical NOT: 1 if a==0, 0 otherwise
*	n/a	a*b	Multiply a and b
/	n/a	a / b	Divide a by b

Operator	XML Equivalent	Syntax	Meaning
+	n/a	a+b	Add a and b
-	n/a	a-b	Subtract b from a
>	>	a > b	1 if a is greater than b, 0 otherwise
<	<	a < b	1 if a is less than b, 0 otherwise
>=	>=	a >= b	1 if a is greater than or equal to b, 0 otherwise
<=	<=	a <= b	1 if a is less than or equal to b, 0 otherwise
==	n/a	a == b	1 if a is equal to b, 0 otherwise
!=	n/a	a != b	1 if a is not equal to b, 0 otherwise
&&	&&	a && b	Logical AND: 1 if both a and b are non-zero, 0 otherwise
	n/a	a b	Logical OR: 1 if either a or b is non-zero, 0 otherwise

value

Specifies the value to be used as criteria for selecting a resource. Value can be numerical, such as when referring to available memory or swap space, or it can be textual, such as when referring to a specific type of host.

Simple selection expression using static resources

The following example selects resources with total memory greater than 500MB:

```
select (maxmem > 500)
```

Compound selection expression using static resources

The following example selects resources with total memory greater than 500MB and total swap space greater than or equal to 300MB:

```
select (maxmem > 500 && maxswp >=300)
```

Compound selection expression with precedence for dynamic load indices

The following example selects resources with available memory greater than 500MB and available swap space greater than or equal to 300MB, or at least 1000MB temporary disk space:

```
select ((mem > 500 && swp >=300) || tmp >= 1000)
```

order

Sorts the selected resources into an order of preference according to the values of the resources.

order synopsis

order(*expression*)

order description

The order string acts on the results of a selection string, sorting the selected resources to identify the most favorable resources, and eliminate the least desirable resources.

Resources are sorted into ascending order based on a load index or the result of an arithmetical expression.

The result orders the resources from smallest to largest.

order expression synopsis

order(*load_index*)

order(*arithmetic_expression*)

load_index

Specify the load index to use as the criteria for sorting resources.

Specify any built-in or external load index.

arithmetic _expression

Specify an arithmetic expression that expresses the desired outcome.

You can create an expression using load indices, operators, and numbers.

Use the following operators:

+	add
-	subtract
*	multiply
/	divide

Order resources based on CPU utilization

The following example orders the selected resources based on their CPU utilization, ordering least utilized hosts first:

```
order(ut)
```

Order resources based on available swap space

The following example orders the selected resources based on their available swap space, but sorts by descending order, ordering hosts with the greatest amount of available swap space first:

```
order(0-swp)
```


-R res_req



Configuration Files

deployment.xml

The configuration file `deployment.xml` defines:

- commands to be run after a service package is deployed
- dependent packages
- environment variables accessible to the package

The file must be called `deployment.xml`, and must be included in the service package.

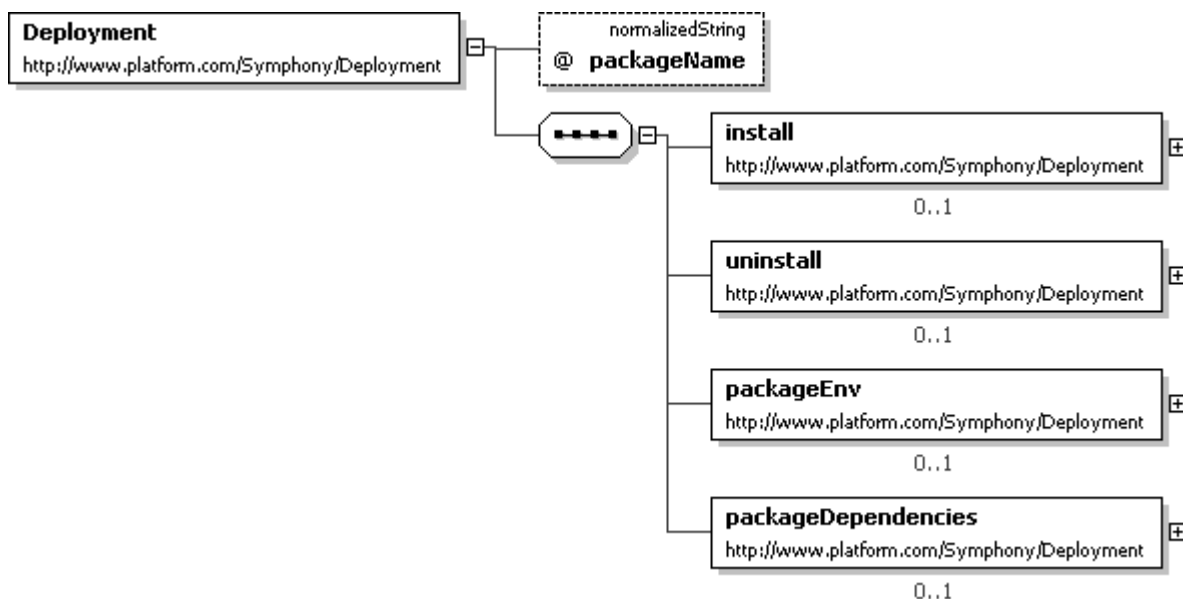
Location

The file must be included at the top level of the service package—it cannot be in a subdirectory. There can only be one file per service package.

Structure

`deployment.xml` has the following sections:

- `install`
- `uninstall`
- `packageEnv`
- `packageDependencies`



Important:

All values in the `deployment.xml` file are case-sensitive when the service is deployed on Linux.

Example

```

<Deployment xmlns="http://www.platform.com/Symphony/Deployment">
  <install>
    <osTypes>
      <osType name="NTX86" startCmd="setup" timeout="600" successCodes="0, 1, 2" />
    
```



```

    <osType name="LINUX86" startCmd="install" timeout="100" successCodes="- 1: 10" />
  </osTypes>
</install>
<uninstall>
  <osTypes>
    <osType name="NTX86" startCmd="setup -u" timeout="30" successCodes=0/>
    <osType name="LINUX86" startCmd="uninstall" timeout="34" successCodes="0" />
  </osTypes>
</uninstall>
<packageEnv>
  <osTypes>
    <osType name="all">
      <env name="PATH">${SOAM_DEPLOY_DIR}/bin</env>
      <env name="env1">val 1</env>
      <env name="env2">val 2</env>
    </osType>
  </osTypes>
</packageEnv>
<packageDependencies>
  <package name="Common" />
</packageDependencies>
</Deployment>
```

install section

Optional. Use the install section to configure commands to run after the package is uncompressed on a compute host.

osType attributes

Attribute	Description	Default Value
name	Required. Operating system type name. The name must match the osType indicated in the application profile, Service section. Default values: <ul style="list-style-type: none">NTX86—For a Windows-based environmentLINUX86—For a Linux-based environment	None

Attribute	Description	Default Value
startCmd	<p>Required.</p> <p>Command to run after the service package is copied on to a compute host and uncompressed.</p> <p>Specify a path relative to the service package installation directory.</p> <p>For example, if your package contained a subdirectory called scripts with the command you want to invoke called myscript, specify:</p> <p>Windows</p> <pre>. \scripts\myscript</pre> <p>Note:</p> <p>To run a Windows .bat script, you need to specify a special syntax. For example:</p> <pre><osType name=NTX86" startCmd="cmd /c cmd /c install.bat" timeout="600" successCodes="0, 1, 2" /></pre> <p>Linux</p> <pre>. /scripts/myscript</pre>	None
timeout	<p>Time that the startCmd is allowed to run before being terminated. Use this attribute to catch a runaway process.</p> <p>The time is counted from the moment the command specified in StartCmd is called.</p>	600 seconds
successCodes	<p>Return codes that indicate successful execution of the startCmd. Specify integers.</p> <p>To specify a list of codes, separate with commas. For example,</p> <pre>successCodes="0, 1"</pre> <p>.</p> <p>To specify a range of codes, separate with a colon. For example, to specify all codes from 0 to 10, enter</p> <pre>successCodes="0: 10"</pre> <p>.</p> <p>Note that if you specify success codes and the code returned is not amongst the specified successCodes, the system considers that the install startCmd failed and deployment is aborted.</p>	Undefined. All return values indicate success.

uninstall section

Optional. Use the uninstall section to configure commands to run if the startCmd specified in the install section fails, or before the package is removed from a compute host.

osType attributes

Attribute	Description	Default Value
name	<p>Required.</p> <p>Operating system type name. The name must match the osType indicated in the application profile, service section.</p> <p>Default values:</p> <ul style="list-style-type: none"> NTX86—For a Windows-based environment LINUX86—For a Linux-based environment 	None
startCmd	<p>Required.</p> <p>Command to run if the startCmd specified in the install section fails, or before the package is removed from a compute host.</p> <p>For example, if your package contained a subdirectory called scripts with the command you want to invoke called myscript, specify:</p> <p>. \scripts\myscript</p> <p>Linux</p> <p>. /scripts/myscript</p>	None
timeout	<p>Time that the startCmd is allowed to run before being terminated. Use this attribute to catch a runaway process.</p> <p>The time is counted from the moment the command specified in StartCmd is called.</p>	600 seconds
successCodes	<p>Codes, which when returned, indicate successful execution of the startCmd.</p> <p>To specify a list of codes, separate with commas. For example, <code>successCodes="0, 1"</code></p> <p>.</p> <p>To specify a range of codes, separate with a colon. For example, to specify all codes from 0 to 10, enter <code>successCodes="0: 10"</code></p> <p>.</p> <p>Note that if you specify success codes and the code returned is not amongst the specified successCodes, the system considers that the uninstall startCmd failed. However, the package is still removed.</p>	Undefined. All return values indicate success.

packageEnv section

Optional. Use the packageEnv section to define environment variables that are accessible to the service package.

osType attribute

Attribute	Description	Default Value
name	<p>Required.</p> <p>Operating system type name. The name must match the osType indicated in the application profile, service section.</p> <p>Default values:</p> <ul style="list-style-type: none"> • NTX86—For a Windows-based environment • LINUX86—For a Linux-based environment 	None

env attribute

Attribute	Description	Default Value
name	<p>Required.</p> <p>Name of the environment variable.</p>	None

packageDependencies section

Optional. Use the packageDependencies section to define packages (such as class libraries) that the service package depends on.

package attribute

Attribute	Description	Default Value
name	<p>Required.</p> <p>Name of the package that the service package depends on.</p>	None

ego.conf

The ego.conf file contains the configuration information for the Symphony cluster. The configuration file is also used to connect to a Symphony cluster from a client machine that is not part of the cluster.

Location

This file is installed with Symphony and is located in %SOAM_HOME%\conf\ on Windows, and in \$SOAM_HOME/conf/ on Linux.

Structure

You cannot change the file name ego.conf.

Note:

To connect to multiple clusters from the same client machine, configure different ego.conf files and rename them to ego.conf when you need to use them.

Parameters

EGO_ADJUST_SHARE_TO_WORKLOAD

Syntax

EGO_ADJUST_SHARE_TO_WORKLOAD=Y | N

Description

Specifies how resources are reclaimed and distributed.

Valid values

Specify one of the following:

Y

Specifies that share ratio is to always be honored when two or more consumers are competing for resources. Whenever consumers compete for resources, resources are reclaimed and distributed in proportion to share ratio. Workload among all consumers is taken into account and resource distribution is adjusted according to workload.

N

When set to n or undefined, resources are distributed according to configured share ratio but resource distribution/reclaim is not adjusted according to workload. When two consumers compete for resources, consumers that have not received up to their share ratio can only reclaim resources up to their share ratio. When consumers have reached their share ratio, distribution of additional resources to consumers is done in First-Come, First-Served order.

EGO_AUDIT_LOG_INHERIT_PERMISSION (Windows only)

Syntax

EGO_AUDIT_LOG_INHERIT_PERMISSION=Y | N

Description

Specifies that all files in the `audit` directory will inherit the permission from the parent (`audit` directory).

Valid values

Specify one of the following:

Y

Specifies that file permission is inherited from the parent directory.

N

Specifies that file permission is not inherited from the parent directory.

EGO_CHAN_KEEPALIVE_TIME

Syntax

EGO_CHAN_KEEPALIVE_TIME=Y | N

Description

Controls how long the TCP connection between management hosts and compute hosts can remain idle (no traffic) before TCP sends a Keep-Alive message.

Valid values

Specify a numeric value of 180 seconds or more.

EGO_DATA_MAXSIZE

Syntax

EGO_DATA_MAXSIZE=*file_size_in_Mbytes*

Description

Limits the maximum size of the allocation event data file (named `ego.stream` by default) where the event logger stores event data. When a data file exceeds this size, the events logger archives the file and creates a new data file. The events logger maintains one archive file and overwrites the old archive with the new archive.

If your system logs a large number of events, you should increase the maximum file size to see more archived event data. If your disk space is insufficient for storing these files, you should decrease the maximum file size, or change the file path to a location with sufficient storage space.

For a production cluster, the average data file switch time, i.e., the time it takes the data file to reach its maximum size and switch to the archive, should be greater than 3 minutes. If the switch time is too frequent, increase the file size. Note that the file size also impacts the data file writing speed so if the file is too large, writing speed will slow down.

ego.conf

Valid values

positive integer

Default

10 Mbytes

EGO_DEFINE_NCPUS

Syntax

EGO_DEFINE_NCPUS=procs | cores | threads

Description

If defined, enables an administrator to define a value other than the number of cores available. Follow one of the three equations below for an accurate value.

- EGO_DEFINE_NCPUS=procs-ncpus=number of processors
- EGO_DEFINE_NCPUS=cores-ncpus=number of processors x number of cores
- EGO_DEFINE_NCPUS=threads-ncpus=number of processors x number of cores x number of threads.

Default

EGO_DEFINE_NCPUS=cores

EGO_DISCIPLINE_TIMEOUT

Syntax

EGO_DISCIPLINE_TIMEOUT=*time_in_seconds*

Description

Adds time to all grace periods in seconds. A grace period is the time the system waits before reclaiming resources from a borrowing consumer when a lending consumer requests them back. This time period allows any running work to finish running before the resource is reclaimed.

This parameter adds time to any grace period, cluster wide. All consumers have a grace period. The grace period's default is 120 seconds.

Default

EGO_DISCIPLINE_TIMEOUT=120

EGO_DISTRIBUTION_INTERVAL

Syntax

EGO_DISTRIBUTION_INTERVAL=*time_in_seconds*

Description

Specifies intervals after which EGO will allocate resources. Requests are queued until the next distribution interval end is reached.

Default

EGO_DISTRIBUTION_INTERVAL=0, EGO will allocate resources as requests for them are made.

EGO_DYNAMIC_HOST_TIMEOUT

Syntax

EGO_DYNAMIC_HOST_TIMEOUT= time_hours | time_minutes

Description

Enables automatic removal of dynamic hosts from the cluster and specifies the timeout value (minimum 10 minutes). To improve performance in very large clusters, you should disable this feature and remove unwanted hosts from the host cache file manually.

Specifies the length of time the system waits for a dynamic host that is unavailable before the master host removes it from the cluster.

Default

Not defined. Unavailable hosts are never removed from the cluster.

Example

EGO_DYNAMIC_HOST_TIMEOUT=60

A dynamic host is removed from the cluster when it is unavailable for 60 hours.

EGO_DYNAMIC_HOST_TIMEOUT=60m

A dynamic host is removed from the cluster when it is unavailable for 60 minutes.

EGO_DYNAMIC_HOST_WAIT_TIME

Syntax

EGO_DYNAMIC_HOST_WAIT_TIME= wait_time | first_wait_time,subsequent_wait_time

Description

When LIM starts on a host, if the master host already recognizes the host (for example, a static host or a dynamic host that had previously started and joined the cluster), it does not need to send a join request to the master LIM. The master LIM sends acknowledgement signals to all hosts that the master host already recognizes.

If LIM does not receive acknowledgement from the master, it will send the join request and wait for the EGO_DYNAMIC_HOST_WAIT_TIME for the acknowledgement again. Therefore, EGO_DYNAMIC_HOST_WAIT_TIME is the interval, in seconds, that the compute host waits for the master LIM acknowledgement so that it can join the cluster.

Once the acknowledgement signal is received, LIM on the compute host will start all other processes such as PEM, ELIM, etc, on the host. If there is no acknowledgement from the master LIM after 20 tries, the local LIM exits, i.e., the cluster join operation fails.

There are two ways to express EGO_DYNAMIC_HOST_WAIT_TIME. You can specify a value that applies to all attempts to join the cluster or you can specify two intervals for EGO_DYNAMIC_HOST_WAIT_TIME. The first interval applies to the first attempt (of the 20 attempts) and the second interval applies to each of the subsequent attempts.

ego.conf

Valid values

positive integer

Default

60 seconds

EGO_ENABLE_BASE_QUOTA

Syntax

EGO_ENABLE_BASE_QUOTA=Y | N

Description

Minimizes the number of reclaims that can occur when consumers are overusing resources from multiple resource groups.

Important:

For this feature to work, you must also enable EGO_ADJUST_SHARE_TO_WORKLOAD in ego.conf.

Valid values

Specify one of the following:

Y

Specifies that resources will be allocated to consumers according to their static share quota from each resource group first, followed by resources allocated according to the order of resource groups defined in ConsumerTrees.xml.

N

Disables this feature.

Default

N (not enabled).

EGO_ENABLE_CHAN_KEEPALIVE

Syntax

EGO_ENABLE_CHAN_KEEPALIVE=Y | N

Description

Determines whether firewall support is enabled between the management hosts and compute hosts. Enabling this feature allows periodic TCP Keep-Alive messages to pass between VEMKD on the management hosts and PEM on the compute hosts.

Restriction:

This parameter is not supported on Solaris management hosts.

Valid values

Specify one of the following:

Y

Enables TCP Keep-Alive messages to pass between management and compute hosts.

N

Disables firewall support.

Default

If the parameter is not set, firewall support is disabled.

EGO_HOST_COMMENT_ENFORCE

Syntax

EGO_HOST_COMMENT_ENFORCE=Y | N

Description

Determines whether a comment is required when requesting to close a host through PMC or CLI.

Valid values

Specify one of the following:

Y

Specifies that a comment is required for the close host operation to succeed.

N

Specifies that a comment is not required for the close host operation.

Default

If the parameter is not set, the comment is not required.

EGO_KD_CLIENT_PORT_RANGE

Syntax

EGO_KD_CLIENT_PORT_RANGE=*port_number1,port_number2*

Description

EGO tries to bind a local port number in the configured range when it works as a client. If there is no available port in the range, EGO uses a random port number allocated by the OS.

Valid values

port_number1 must be greater than 1024 and *port_number2* must be greater than *port_number1*.

ego.conf

EGO_KD_PORT

Syntax

```
EGO_KD_PORT=port_number
```

Description

Specifies the port number to use to connect to the Symphony cluster.

Valid values

The port number must exactly match the port number specified in the master `ego.conf` in the cluster.

Default

If the port is not specified, it defaults to 7870.

EGO_MASTER_LIST

Syntax

```
EGO_MASTER_LIST="master_candidate1 master_candidate2 . . ."
```

Description

Specifies the hosts that are master host candidates in the cluster to which you want to connect.

Valid values

The host names indicated here must exactly match the host names specified in the master `ego.conf`, accessible from the master host under `%EGO_CONFDIR%` on Windows and `$EGO_CONFDIR` on Linux.

Specify a list of hosts separated by spaces.

EGO_PARENT_QUOTA

Syntax

```
EGO_PARENT_QUOTA=Y | N
```

Description

Modifies the allocation and reclaim behavior of consumers so that, when a child consumer releases its allocation to the free pool, any of its siblings that have a demand for resources gets them before other consumers do.

Default

```
EGO_PARENT_QUOTA=N
```

EGO_SEC_PLUGIN

Syntax

```
EGO_SEC_PLUGIN=sec_ego_default | external_plugin
```


Description

Specifies the security mechanism to use when connecting to the Symphony cluster.

Valid values

The value must exactly match the value specified in the actual `ego.conf` in the cluster. Specify one of the following:

`sec_ego_default`

Specifies to use the default authentication.

`external_plugin`

Specifies to use an external, third-party authentication.

Example

```
EGO_MASTER_LIST="hosta hostc hostd"
EGO_KD_PORT=7870
EGO_SEC_PLUGIN=sec_ego_default
```

EGO_STRIP_DOMAIN

Syntax

`EGO_STRIP_DOMAIN=domain_suffix[:domain_suffix ...]`

Description

(Optional) If all of the hosts in your cluster can be reached using short host names, you can configure EGO to use the short host names by specifying the portion of the domain name to remove. If your hosts are in more than one domain or have more than one domain name, you can specify more than one domain suffix to remove, separated by a colon (:).

For example, given this definition of `EGO_STRIP_DOMAIN`,

```
EGO_STRIP_DOMAIN=.foo.com:.bar.com
```

EGO accepts `hostA`, `hostA.foo.com`, and `hostA.bar.com` as names for host `hostA`, and uses the name `hostA` in all output. The leading period '.' is required.

Example:

```
EGO_STRIP_DOMAIN=.platform.com:.generic.com
```

In the above example, EGO accepts `hostA`, `hostA.platform.com`, and `hostA.generic.com` as names for host `A`, and uses the name `hostA` in all output.

Setting this parameter only affects host names displayed through EGO, it does not affect DNS host lookup.

Default

Not defined

vem_resource.conf

Defines the hosts in the Symphony DE cluster. This file is not used with Symphony on the grid.

Location

The `vem_resource.conf` configuration file exists on each host in the SymphonyDE cluster, including the management host, and all compute hosts.

The default locations for `vem_resource.conf` are

- Windows—%SOAM_HOME%\conf
- Linux—\$SOAM_HOME/conf

Structure

Each line defines a single host.

The `vem_resource.conf` file must be updated on each host whenever hosts are added or removed, or when the configuration parameters of any host in the network changes.

There are five types of hosts you configure: session manager, compute host, session director, repository service, and GUI service.

Session manager host:

AGENT: *port_number: host_name: max_SSMs: 0: osType: CPU_factor*

Compute host:

AGENT: *port_number: host_name: 0: max_SIMs: osType: CPU_factor*

Session director host:

SD_SDK: *port_number: host_name: sd_startCmd*

SD_ADMIN: *port_number: host_name: sd_startCmd*

Repository service host:

RS_DEPLOY: *port_number: host_name: rs_startCmd*

GUI service host:

WEBGUI: *port_number: host_name: startguiservice*

Attributes

port_number

For AGENT, specifies the port number the Platform EGO emulator (`start_agent`) uses. The `start_agent` process runs on each Symphony host.

For SD_SDK, SD_ADMIN, specifies port numbers used by the session director.

For RS_DEPLOY, specifies the port number used by the repository service.

For WEBGUI, specifies the port number used by the GUI service. Port number for WEBGUI is 18080.

host_name

Specifies localhost in single-host environment.

In a multi-host environment, specify using dotted-decimal notation or as a DNS name. Hosts can also have dynamic IP addresses (DHCP).

max_SSMs

Specifies the maximum number of session managers that can be started on this host. For a compute host, specify **0**.

max_SIMs

Specifies the maximum number of service instance managers that can be started on this host. Each service instance manager requires one (virtual) slot to run on, therefore the maximum number of sims the compute host can run is equal to the number of CPU slots on the machine. For a dedicated session manager host, specify **0** to prevent work from running on this host.

osType

Specifies the operating system type the host runs:

- **LINUX86**—A Linux-based environment, such as RedHat Linux
- **NTX86**—A Windows-based environment, such as Windows 2003, Windows 2000, Windows NT, and Windows XP

CPU_factor

Default=1.

The CPU factor is the speed of the host's CPU relative to other hosts in the cluster. If one processor is twice the speed of another, its CPU factor should be twice as large. CPU factors are defined by the cluster administrator. For multiprocessor hosts, the CPU factor is the speed of a single processor.

sd_startCmd

Specifies the command that starts the session director.

Specify **sd** unless you have changed the name of the command or moved the session director executable from the installation directory. In this case, specify the absolute path to the session director executable.

rs_startCmd

Specifies the name of the command that starts the repository service.

Specify **rs** unless you have changed the name of the command or moved the repository service executable from the installation directory. In this case, specify the absolute path to the repository director executable.

GUI_startcmd

Specifies the name of the command that starts the WebGUI service.

Specify **startgui service** unless you have changed the name of the command or moved the startguiservice script from the installation directory. In this case, specify the absolute path to the startguiservice script.

SOAM_SESSION_NAME_COMPATIBILITY

This is a flag to specify that there is no limitation on characters for the session name, i.e., any characters can be used. If this attribute is not specified, only alphanumeric, '-', and '_' characters are allowed in the session name.

Examples

Management host also runs workload

In the following example, host1 runs the session director, repository service, GUI service, five session manager processes, and five service instance manager processes per application.

```
# Resource configuration file
#
# File format:
# <service name>: <port_number>: <host_name>: <max number of SSMS SD can
start>: <max number of SIMs SSM can start>: <OS_type>: <CPU_factor>
#
AGENT:8000:host1:5:5:LINUX86:1
#
# SD service information
# <service name>: <port_number>: <host_name>: <sd startcmd>
SD_SDK:15051:host1:sd
SD_ADMIN:15050:host1:sd
#
# RS service information
# <service name>: <port_number>: <host_name>: <rs startcmd>
RS_DEPLOY:15052:host1:rs
#
# GUI service information
# <service name>: <port_number>: <host_name>: <gui startcmd>
# Note: The port number of WEBGUI is fixed as 18080.
WEBGUI:18080:host1:startguiservice
```

One management host and one compute host

In the following example, the host myfirsthost runs the session director, the repository service, five session manager processes and GUI service. The compute host, mysecondhost runs five service instance manager processes per application.

```
# Resource configuration file
#
# File format:
# <service name>: <port_number>: <host_name>: <max number of SSMS SD can
start>: <max number of SIMs SSM can start>: <OS_type>: <CPU_factor>
#
AGENT:8000:myfirsthost:5:0:LINUX86:1
AGENT:8000:mysecondhost:0:5:LINUX86:1
#
# SD service information
# <service name>: <port_number>: <host_name>: <sd startcmd>
SD_SDK:15051:myfirsthost:sd
SD_ADMIN:15050:myfirsthost:sd
#
# RS service information
# <service name>: <port_number>: <host_name>: <rs startcmd>
RS_DEPLOY:15052:myfirsthost:rs
# WebGUI service information
# <service name>: <port_number>: <host_name>: <gui startcmd>
# Note: The port number of WEBGUI is fixed as 18080.
WEBGUI:18080:myfirsthost:startguiservice
```


ego.sudoers

Contents

- About ego.sudoers
- The ego.sudoers file
- File format
- Creating and modifying ego.sudoers
- Parameters

About ego.sudoers

The `ego.sudoers` file is an optional file to configure security mechanisms. It is not installed by default.

You use `ego.sudoers` to grant permission to users other than root to perform certain operations as root in EGO.

The parameters in this file apply to UNIX hosts only. They are not required for Windows hosts because all users with membership in the `Platform Services Admin` group can start EGO daemons.

If `ego.sudoers` does not exist, only root can perform these operations in EGO on UNIX.

The ego.sudoers file

In EGO, certain operations such as daemon startup can only be performed by root. The `ego.sudoers` file grants root privileges to specific users or user groups to perform these operations.

Location

`ego.sudoers` must be located in `/etc` on each host.

Permissions

`ego.sudoers` must have permission 600 and be readable and writable only by root.

File format

Each entry can have one of the following forms:

- `NAME=VALUE`
- `NAME=`
- `NAME="STRING1 STRING2 . . ."` except for the parameter `EGO_STARTUP_ALTERNATE_PATHS`, which has the format `NAME=STRING1: STRING2 . . .`

The equal sign `=` must follow each NAME even if no value follows and there should be no space beside the equal sign.

NAME describes an authorized operation.

VALUE is a single string or multiple strings. The value for `EGO_STARTUP_USERS` is separated by spaces and enclosed in quotation marks. The value for `EGO_STARTUP_ALTERNATE_PATHS` is separated by colons.

Example ego.sudoers file

```
EGO_STARTUP_PATH=/usr/share/ego/etc EGO_STARTUP_ALTERNATE_PATHS=/usr/share/ego_cluster_1/ego/
1.2/ai x5-64/etc:/usr/share/ego_cluster_2/ego/1.2/ai x5-64/etc EGO_STARTUP_USERS="user1 user10
user55"
```

Creating and modifying ego.sudoers

You can modify `ego.sudoers` with a text editor if you need to specify an alternate path or paths for a parallel installation.

This file enables the EGO daemon startup control feature when `EGO_STARTUP_USERS` is also defined. Define both parameters when you want to allow users other than `root` to start EGO daemons.

Parameters

- `EGO_STARTUP_PATH`
- `EGO_STARTUP_ALTERNATE_PATHS`
- `EGO_STARTUP_USERS`

EGO_STARTUP_PATH

Syntax

EGO_STARTUP_PATH=*path*

Description

Specifies the absolute path name of the directory in which the EGO daemon binary files (`lim`, `vemkd`, and `egosc`) are installed. EGO daemons are usually installed in the path specified by `EGO_SERVERDIR` defined in the `cschrc.ego` or `profile.ego` files.

Default

Not defined. Only the `root` user account can start EGO daemons.

EGO_STARTUP_ALTERNATE_PATHS

Syntax

EGO_STARTUP_ALTERNATE_PATHS=*path:path...*

Description

For parallel installations and clusters, provides alternate paths to control multiple clusters. Define both parameters when you have multiple parallel installation paths to the directories of the EGO daemon binary files (`lim`, `vemkd`, and `egosc`) and want to allow users other than `root` to start EGO daemons.

EGO daemons are usually installed in the path specified by `EGO_SERVERDIR` defined in the `cschrc.ego` or `profile.ego` files.

The maximum length of the path string is 4000 characters.

Default

Not defined. Only the `root` user account can start EGO daemons.

EGO_STARTUP_USERS

Syntax

EGO_STARTUP_USERS=all_admins | "*user_name...*"

Description

Enables the EGO daemon startup control feature when EGO_STARTUP_PATH is also defined. Define both parameters when you want to allow users other than root to start EGO daemons.

On UNIX hosts, by default only root can start EGO daemons. To manually start EGO daemons, a user runs the command egosh, which has been made setuid root by the egosetsudoers script.

EGO_STARTUP_USERS specifies a list of user accounts that can successfully run the command egosh to start EGO daemons.

all_admins

- Allows all UNIX users defined as EGO administrators in the file `ego.cluster.cluster_name` to start EGO daemons as root by running the egosh command.
- Not recommended due to the security risk of a non-root EGO administrator adding to the list of administrators in the `ego.cluster.cluster_name` file.

"user_name..."

- Allows the specified user accounts to start EGO daemons by running the egosh command.
- Separate multiple user names with a space.
- For a single user, do not use quotation marks.

Default

Not defined. Only the root user account can start EGO daemons.

ego.*cluster_name*.license.acct

This is the accounting file containing EGO license information. There is one for each cluster, called ego. *cluster_name*.license.acct. The *cluster_name* variable is the name of the cluster defined in the Cluster section of ego.shared.

ego.*cluster_name*.license.acct structure

The license audit log file is an ASCII file with one record per line. The fields of a record are separated by blanks.

File properties

Location

The default location of this file is defined by EGO_LOGDIR in ego.conf, but you can override this by defining EGO_LICENSE_ACCT_PATH in ego.conf.

Owner

The primary EGO admin is the owner of this file.

Permissions

```
-rw-r--r--
```

Records and fields

The fields of a record are separated by blanks. The fields in order of occurrence are as follows:

timestamp

Time stamp of the logged event (in seconds since the epoch).

feature-type

Valid value: EGO_BASE

feature-version

The version of the EGO product.

value

The actual tracked value.

Format:

peak max

Where

peak is the peak usage value (in number of CPUs).

max is the maximum availability and usage values (in number of CPUs).

status

The results of the license usage check. The only valid value is OK, which means peak usage is less than the maximum license availability.

hash

Line encryption used to authenticate the record.

Example record Format

```
1249592702 EGO_BASE 1. 2 1 2147483647 OK d18372042d44084aa68279400c95c7bacf26be9d
```


ego.shared

The `ego. shared` file contains common definitions that are shared by Symphony clusters defined by `ego. cluster. cluster_name` files. This includes lists of cluster names, host types, host models, the special resources available, and external load indices.

This file is installed by default in the directory defined by `EGO_CONFDIR`.

Changing ego.shared configuration

After making any changes to `ego. shared`, run the following command:

```
egosh ego restart
```

Cluster section

(Required) Lists the cluster names recognized by Symphony.

Cluster section structure

The first line must contain the mandatory keyword `ClusterName`. The other keyword is optional.

Each subsequent line defines one cluster.

Example Cluster section

```
Begin Cluster
ClusterName # Keyword
cluster1
End Cluster
```

ClusterName

Defines all cluster names recognized by Symphony.

The cluster name referenced anywhere by Symphony must be defined here. The file names of cluster-specific configuration files must end with the associated cluster name.

HostType section

(Required) Lists the valid host types in the cluster. All hosts that can run the same binary executable are in the same host type.

HostType section structure

The first line consists of the mandatory keyword `TYPENAME`.

Subsequent lines name valid host types.

Example HostType section

```

Begin HostType
  TYPENAME
  SOL64
  SOLSPARC
  LINUX86LINUXPPC
  LINUX64
  NTX86
  NTX64
  NTIA64
End HostType

```

TYPENAME

Host type names are usually based on a combination of the hardware name and operating system. If your site already has a system for naming host types, you can use the same names for Symphony.

HostModel section

(Required) Lists models of machines and gives the relative CPU scaling factor for each model. All hosts of the same relative speed are assigned the same host model.

Symphony uses the relative CPU scaling factor to normalize the CPU load indices so that tasks are more likely to be sent to faster hosts. The CPU factor affects the calculation of task execution time limits and accounting. Using large or inaccurate values for the CPU factor can cause confusing results when CPU time limits or accounting are used.

HostModel section structure

The first line consists of the mandatory keywords **MODELNAME**, **CPUFACTOR**, and **ARCHITECTURE**.

Subsequent lines define a model and its CPU factor.

Example HostModel section

```

Begin HostModel  MODELNAME  CPUFACTOR  ARCHITECTURE
PC400           13.0      (i86pc_400 i686_400)
PC450           13.2      (i86pc_450 i686_450)
Sparc5F         3.0      (SUNWSPARCstation5_170_sparc)
Sparc20         4.7      (SUNWSPARCstation20_151_sparc)
Ultra5S        10.3      (SUNWUltra5_270_sparcv9 SUNWUltra510_270_sparcv9)
End HostModel

```

ARCHITECTURE

(Reserved for system use only) Indicates automatically detected host models that correspond to the model names.

CPUFACTOR

Though it is not required, you would typically assign a CPU factor of 1.0 to the slowest machine model in your system and higher numbers for the others. For example, for a machine model that executes at twice the speed of your slowest model, a factor of 2.0 should be assigned.

MODELNAME

Generally, you need to identify the distinct host types in your system, such as MIPS and SPARC first, and then the machine models within each, such as SparcIPC, Sparc1, Sparc2, and Sparc10.

About automatically detected host models and types

When you first install Symphony, you do not necessarily need to assign models and types to hosts in `ego.cluster.cluster_name`. If you do not assign models and types to hosts in `ego.cluster.cluster_name`, LIM automatically detects the model and type for the host.

Automatic detection of host model and type is useful because you no longer need to make changes in the configuration files when you upgrade the operating system or hardware of a host and reconfigure the cluster. Symphony will automatically detect the change.

Mapping to CPU factors

Automatically detected models are mapped to the short model names in `l sf.shared` in the ARCHITECTURE column. Model strings in the ARCHITECTURE column are only used for mapping to the short model names.

Example `ego.shared` file:

Begin HostModel

MODELNAME	CPUFACTOR	ARCHITECTURE
SparcU5	5.0	(SUNWUltr510_270_sparcv9)
PC486	2.0	(i486_33 i486_66)
PowerPC	3.0	(PowerPC12 PowerPC16 PowerPC31)

End HostModel

If an automatically detected host model cannot be matched with the short model name, it is matched to the best partial match and a warning message is generated.

If a host model cannot be detected or is not supported, it is assigned the DEFAULT model name and an error message is generated.

Naming convention

Models that are automatically detected are named according to the following convention:

hardware_platform [*_processor_speed* [*_processor_type*]]

where:

- *hardware_platform* is the only mandatory component
- *processor_speed* is the optional clock speed and is used to differentiate computers within a single platform
- *processor_type* is the optional processor manufacturer used to differentiate processors with the same speed
- Underscores (`_`) between *hardware_platform*, *processor_speed*, *processor_type* are mandatory.

Resource section

Optional. Defines resources (must be done by the cluster administrator).

Resource section structure

The first line consists of the keywords. RESOURCENAME and DESCRIPTION are mandatory. The other keywords are optional. Subsequent lines define resources.

Example Resource section

```

Begin Resource
RESOURCENAME  TYPE      INTERVAL  INCREASING  DESCRIPTION  # Keywords
fs            Boolean ()      ()          (File server)
cs            Boolean ()      ()          (Compute server)
frame        Boolean ()      ()          (Hosts with FrameMaker licence)
bigmem       Boolean ()      ()          (Hosts with very big memory)
diskless     Boolean ()      ()          (Diskless hosts)
linux        Boolean ()      ()          (LINUX UNIX)
nt           Boolean ()      ()          (Windows NT)
mg           Boolean ()      ()          (Management hosts)
scode        Numeric 5      Y          (Host scavenging code)
scvg         Boolean ()      ()          (Resource tag identifying scavenge-capable
                                hosts)

agent_control
  String 5      ()          (Host scavenging flag)
cit       Numeric 5      N          (Amount of time in minutes that a CPU has
                                been idle)
uit_t     Numeric 5      Y          (Idle time threshold, in minutes)
cu_t      Numeric 5      Y          (Adjusted CPU utilization threshold, as a
                                percentage)
cit_t     Numeric 5      Y          (CPU idle time threshold, in minutes)
define_ncpus_procs
  Boolean ()      ()          (ncpus := procs)
define_ncpus_cores
  Boolean ()      ()          (ncpus := cores)
define_ncpus_threads
  Boolean ()      ()          (ncpus := threads)
svrscvg    Boolean ()      ()          (Resource tag identifying server scavenge
                                capable hosts)
vmscvg     Boolean ()      ()          (Resource tag identifying harvesting scavenge
                                capable hosts)
acu        Numeric 5      Y          (Adjusted CPU utilization which not include
                                CPU usage of symphony and exempt process
                                list, as a percentage)

exempt_process
  String 5      ()          (process list which will be excluded for
                                calculating CPU usage)
close_process
  String 5      ()          (process list which will trigger host close
                                or not open)

End Resource

```

RESOURCENAME

The name you assign to the new resource. An arbitrary character string.

- A resource name cannot begin with a number.
- A resource name cannot contain any of the following characters:

```
: . ( ) [ + - * / ! & | < > @ =
```

- A resource name cannot be any of the following reserved names:

```

cpu cpuf io logins ls idle maxmem maxswp maxtmp type model status it
mem ncpus define_ncpus_cores define_ncpus_procs
define_ncpus_threads ndisks pg r15m r15s r1m swap swp tmp ut

```


- To avoid conflict with `inf` and `nan` keywords in 3rd-party libraries, resource names should not begin with `inf` or `nan` (upper case or lower case). Resource requirement strings, such as `-R "infra"` or `-R "nano"` will cause an error. Use `-R "defined(infix)"` or `-R "defined(nanxx)"`, to specify these resource names.
- Resource names are case sensitive
- Resource names can be up to 39 characters in length
- For Solaris machines, the keyword `int` is reserved and cannot be used.

TYPE

The type of resource:

- Boolean—Resources that have a value of 1 on hosts that have the resource and 0 otherwise.
- Numeric—Resources that take numerical values, such as all the load indices, number of processors on a host, or host CPU factor.
- String—Resources that take string values, such as host type, host model, host status.

Default

If TYPE is not given, the default type is Boolean.

INTERVAL

Optional. Applies to dynamic resources only.

Defines the time interval (in seconds) at which the resource is sampled by the ELIM.

If INTERVAL is defined for a numeric resource, it becomes an external load index.

Default

If INTERVAL is not given, the resource is considered static.

INCREASING

Applies to numeric resources only.

If a larger value means greater load, INCREASING should be defined as Y. If a smaller value means greater load, INCREASING should be defined as N.

DESCRIPTION

Brief description of the resource.

ego.cluster

- About ego.cluster
- Parameters section
- ClusterAdmins section
- Host section
- ResourceMap section

Changing ego.cluster configuration

After making any changes to `ego.cluster.cluster_name`, run the following command to restart the master host:

```
egosh ego restart master_host
```

Location

This file is typically installed in the directory defined by `EGO_CONFDIR`.

Structure

The `ego.cluster.cluster_name` file contains the following configuration sections:

- Parameters section
- ClusterAdmins section
- Host section
- ResourceMap section

Parameters section

About ego.cluster

This is the cluster configuration file. There is one for each cluster, called `ego.cluster.cluster_name`. The `cluster_name` suffix is the name of the cluster defined in the Cluster section of `ego.shared`. All Symphony hosts are listed in this file, along with the list of Symphony administrators and the installed Symphony features.

The `ego.cluster.cluster_name` file contains configuration information that affects all Symphony applications. It defines cluster administrators, hosts that make up the cluster, attributes of each individual host such as host type or host model, and resources using the names defined in `ego.shared`.

Parameters

- ELIM_POLL_INTERVAL
- EGO_HOST_ADDR_RANGE
- PRODUCTS

ELIM_POLL_INTERVAL

Syntax

```
ELIM_POLL_INTERVAL=seconds
```


Description

Time interval, in seconds, that the LIM samples external load index information. If your `elim` executable is programmed to report values more frequently than every 5 seconds, set the `ELIM_POLL_INTERVAL` so that it samples information at a corresponding rate.

For host scavenging with FastRelease mode, configure `ELIM_POLL_INTERVAL=1` for the the fastest response time.

Valid values

1 to 5

Default

5 seconds

EGO_HOST_ADDR_RANGE

Syntax

`EGO_HOST_ADDR_RANGE=IP_address ...`

Description

Identifies the range of IP addresses that are allowed to be Symphony hosts, which can be dynamically added to or removed from the cluster.

Caution:

To enable dynamically added hosts after installation, you must define `EGO_HOST_ADDR_RANGE` in `ego.cluster.cluster_name`, and `EGO_DYNAMIC_HOST_WAIT_TIME` in `ego.conf`. If you enable dynamic hosts during installation, you must define an IP address range after installation to enable security.

If a value is defined, security for dynamically adding and removing hosts is enabled, and only hosts with IP addresses within the specified range can be added to or removed from a cluster dynamically.

Specify an IP address or range of addresses, using either a dotted quad notation (IPv4) or IP Next Generation (IPv6) format. Symphony supports both formats; you do not have to map IPv4 addresses to an IPv6 format. Multiple ranges can be defined, separated by spaces.

Note:

To use IPv6 addresses, you must define the parameter `EGO_ENABLE_SUPPORT_IPV6` in `ego.conf`.

If there is an error in the configuration of `EGO_HOST_ADDR_RANGE` (for example, an address range is not in the correct format), no host will be allowed to join the cluster dynamically and an error message will be logged in the LIM log. Address ranges are validated at startup, reconfiguration, or restart, so they must conform to the required format.

If a requesting host belongs to an IP address that falls in the specified range, the host will be accepted to become a dynamic Symphony host.

IP addresses are separated by spaces, and considered "OR" alternatives.

If you define the parameter `EGO_HOST_ADDR_RANGE` with:

- No range specified, all IPv4 and IPv6 clients are allowed.
- Only an IPv4 range specified, only IPv4 clients within the range are allowed.
- Only an IPv6 range specified, only IPv6 clients within the range are allowed.
- Both an IPv6 and IPv4 range specified, IPv6 and IPv4 clients within the ranges are allowed.

The asterisk (*) character indicates any value is allowed.

The dash (-) character indicates an explicit range of values. For example 1-4 indicates 1,2,3,4 are allowed.

Open ranges such as *-30, or 10-*, are allowed.

For IPv6 addresses, the double colon symbol (::) indicates multiple groups of 16-bits of zeros. You can also use (::) to compress leading and trailing zeros in an address filter, as shown in the following example:

`EGO_HOST_ADDR_RANGE=1080::8:800:20fc:*`

This definition allows hosts with addresses 1080:0:0:0:8:800:20fc:* (three leading zeros).

You cannot use the double colon (::) more than once within an IP address. You cannot use a zero before or after (::). For example, 1080:0::8:800:20fc:* is not a valid address.

If a range is specified with fewer fields than an IP address such as 10.161, it is considered as 10.161.*.*.

This parameter is limited to 2048 characters.

Notes

After you configure `EGO_HOST_ADDR_RANGE`, check the `lim.1og. host_name` file to make sure this parameter is correctly set. If this parameter is not set or is wrong, this will be indicated in the log file.

Examples

`EGO_HOST_ADDR_RANGE=100`

All IPv4 and IPv6 hosts with a domain address starting with 100 will be allowed access.

- To specify only IPv4 hosts, set the value to **`100.*`**
- To specify only IPv6 hosts, set the value to **`100:*`**

`EGO_HOST_ADDR_RANGE=100-110.34.1-10.4-56`

All hosts belonging to a domain with an address having the first number between 100 and 110, then 34, then a number between 1 and 10, then, a number between 4 and 56 will be allowed access. No IPv6 hosts are allowed. Example: 100.34.9.45, 100.34.1.4, 102.34.3.20, etc.

`EGO_HOST_ADDR_RANGE=100.172.1.13 100.*.30-54 124.24-*.1.*-34`

The host with the address 100.172.1.13 will be allowed access. All hosts belonging to domains starting with 100, then any number, then a range of 30 to 54 will be allowed access. All hosts belonging to domains starting with 124, then from 24 onward, then 1, then from 0 to 34 will be allowed access. No IPv6 hosts are allowed.

`EGO_HOST_ADDR_RANGE=12.23.45.*`

All hosts belonging to domains starting with 12.23.45 are allowed. No IPv6 hosts are allowed.

`EGO_HOST_ADDR_RANGE=100.*43`

The * character can only be used to indicate any value. The format of this example is not correct, and an error will be inserted in the LIM log and no hosts will be able to join the cluster dynamically. No IPv6 hosts are allowed.


```
EGO_HOST_ADDR_RANGE=100.*43 100.172.1.13
```

Although one correct address range is specified, because *43 is not correct format, the entire line is considered not valid. An error will be inserted in the LIM log and no hosts will be able to join the cluster dynamically. No IPv6 hosts are allowed.

```
EGO_HOST_ADDR_RANGE = 3ffe
```

All client IPv6 hosts with a domain address starting with 3ffe will be allowed access. No IPv4 hosts are allowed.

```
EGO_HOST_ADDR_RANGE = 3ffe:fffe::88bb:*
```

Expands to 3ffe:fffe:0:0:0:0:88bb:*. All IPv6 client hosts belonging to domains starting with 3ffe:fffe::88bb:* are allowed. No IPv4 hosts are allowed.

```
EGO_HOST_ADDR_RANGE = 3ffe-4fff:fffe::88bb:aa-ff 12.23.45.*
```

All IPv6 client hosts belonging to domains starting with 3ffe up to 4fff, then fffe::88bb, and ending with aa up to ff are allowed. IPv4 client hosts belonging to domains starting with 12.23.45 are allowed.

```
EGO_HOST_ADDR_RANGE = 3ffe-*:fffe::88bb:*-ff
```

All IPv6 client hosts belonging to domains starting with 3ffe up to ffff and ending with 0 up to ff are allowed. No IPv4 hosts are allowed.

Default

Undefined (dynamic host feature disabled). If you enable dynamic hosts during installation, no security is enabled and all hosts can join the cluster.

See also

```
EGO_ENABLE_SUPPORT_IPV6
```

EXINTERNAL

Syntax

```
EXINTERNAL=time_in_seconds
```

Description

Time interval, in seconds, at which the LIM daemons exchange load information.

On extremely busy hosts or networks, or in clusters with a large number of hosts, load may interfere with the periodic communication between LIM daemons. Setting EXINTERVAL to a longer interval can reduce network load and slightly improve reliability, at the cost of slower reaction to dynamic load changes.

Note that if you define the time interval as less than 5 seconds, EGO automatically resets it to 5 seconds.

Default

15 seconds

PRODUCTS

Syntax

PRODUCTS=*product_keyword* ...

Description

Specifies the Symphony products that the cluster will run (you must also have a license for each product). The list of items is separated by space.

The PRODUCTS parameter is set automatically during Symphony installation to include LSF_Base, which is required to run Symphony. Specify additional product keywords if your cluster is fully licensed for the corresponding products.

For partially licensed products, do not include the product keyword in this parameter, configure the RESOURCES parameter in the Hosts section of this file instead.

Valid values

Default

LSF_Base

ClusterAdmins section

(Optional) The ClusterAdmins section defines the Symphony administrators for the cluster. The only keyword is ADMINISTRATORS.

If the ClusterAdmins section is not present, the default Symphony administrator is root. Using root as the primary Symphony administrator is not recommended.

ADMINISTRATORS

Syntax

ADMINISTRATORS=*administrator_name* ...

Description

Specify UNIX user names.

You can also specify UNIX user group names, Windows user names, and Windows user group names. To specify a Windows user account or user group, include the domain name in uppercase letters (*DOMAIN_NAME**user_name* or *DOMAIN_NAME**user_group*).

The first administrator of the expanded list is considered the primary Symphony administrator. The primary administrator is the owner of the EGO configuration files, as well as the working files under *EGO_SHAREDIR/cluster_name*. If the primary administrator is changed, make sure the owner of the configuration files and the files under *EGO_SHAREDIR/cluster_name* are changed as well.

Administrators other than the primary Symphony administrator have the same privileges as the primary Symphony administrator except that they do not have permission to change EGO configuration files. They can perform clusterwide operations on jobs, queues, or hosts in the system.

For flexibility, each cluster may have its own Symphony administrators, identified by a user name, although the same administrators can be responsible for several clusters.

Windows domain:

- If the specified user or user group is a domain administrator, member of the Power Users group or a group with domain administrative privileges, the specified user or user group must belong to the Symphony user domain.
- If the specified user or user group is a user or user group with a lower degree of privileges than outlined in the previous point, the user or user group must belong to the Symphony user domain and be part of the Global Admins group.

Windows workgroup

- If the specified user or user group is not a workgroup administrator, member of the Power Users group, or a group with administrative privileges on each host, the specified user or user group must belong to the Local Admins group on each host.

Example

The following gives an example of a cluster with two Symphony administrators. The user listed first, user2, is the primary administrator.

```
Begin ClusterAdmins
ADMINISTRATORS = user2 user7
End ClusterAdmins
```

Default

egoadmin

Host section

The Host section is the last section in `ego.cluster`. *cluster_name* and is the only required section. It lists all the hosts in the cluster and gives configuration information for each host.

The order in which the hosts are listed in this section is important, because the first host listed becomes the Symphony master host. Since the master LIM makes all placement decisions for the cluster, it should be on a fast machine.

The LIM on the first host listed becomes the master LIM if this host is up; otherwise, the second host becomes the master if it is up, and so on. Also, to avoid the delays involved in switching masters if the first machine goes down, the master should be on a reliable machine. It is desirable to arrange the list such that the first few hosts in the list are always in the same subnet. This avoids a situation where the second host takes over as master when there are communication problems between subnets.

Configuration information is of two types:

- Some fields in a host entry simply describe the machine and its configuration.
- Other fields set thresholds for various resources.

Example Host section

This example Host section contains descriptive and threshold information for three hosts:

Begin Host								
HOSTNAME	model	type	server	r1m	pg	tmp	RESOURCES	RUNWINDOW
hostA	SparcIPC	Sparc	1	3.5	15	0	(sunos frame)	()
hostD	Sparc10	Sparc	1	3.5	15	0	(sunos)	(5: 18: 30- 1: 8: 30)
hostD	!	!	1	2.0	10	0	()	()
hostE	!	!	1	2.0	10	0	(linux !bigmem)	()
End Host								

Descriptive fields

The following fields are required in the Host section:

- HOSTNAME
- RESOURCES
- type
- model

The following fields are optional:

- server
- nd
- RUNWINDOW
- REXPRI

HOSTNAME

Description

Official name of the host as returned by `hostname(1)`

The name must be listed in `ego.shared` as belonging to this cluster.

model

Description

Host model

The name must be defined in the HostModel section of `ego.shared`. This determines the CPU speed scaling factor applied in load and placement calculations.

Optionally, the `!` keyword for the model or type column, indicates that the host model or type is to be automatically detected by the LIM running on the host.

RESOURCES

Description

The static Boolean resources and static or dynamic numeric and string resources available on this host.

The resource names are strings defined in the Resource section of `ego. shared`. You may list any number of resources, enclosed in parentheses and separated by blanks or tabs. For example:

```
(fs frame hpux)
```

Optionally, you can specify an exclusive resource by prefixing the resource with an exclamation mark (!). For example, resource `bi gmem` is defined in `ego. shared`, and is defined as an exclusive resource for `hostE`:

```
Begin Host
HOSTNAME  model    type    server r1m pg tmp RESOURCES          RUNWI NDOW
...
hostE     !        !        1      2.0 10   0 (linux !bi gmem)  ()
...
End Host
```

Square brackets are not valid and the resource name must be alphanumeric.

You can specify static and dynamic numeric and string resources in the resource column of the Host clause. For example:

```
Begin Host
HOSTNAME  model  type  server r1m  mem  swp RESOURCES  #Keywords
hostA     !     !     1      3.5  ()   ()  (mg elimres patchrev=3 owner=user1)
hostB     !     !     1      3.5  ()   ()  (specman=5 switch=1 owner=test)
hostC     !     !     1      3.5  ()   ()  (switch=2 rack=rack2_2_3 owner=test)
hostD     !     !     1      3.5  ()   ()  (switch=1 rack=rack2_2_3 owner=test)
End Host
```

type

Description

Host type as defined in the HostType section of `ego. shared`

The strings used for host types are determined by the system administrator: for example, `SUNSOL`, `DEC`, or `HPPA`. The host type is used to identify binary-compatible hosts.

The host type is used as the default resource requirement. That is, if no resource requirement is specified in a placement request, the task is run on a host of the same type as the sending host.

Often one host type can be used for many machine models. For example, the host type name `SUNSOL6` might be used for any computer with a SPARC processor running SunOS 6. This would include many Sun models and quite a few from other vendors as well.

Optionally, the `!` keyword for the model or type column, indicates that the host model or type is to be automatically detected by the LIM running on the host.

Threshold fields

The LIM uses these thresholds in determining whether to place remote jobs on a host. If one or more Symphony load indices exceeds the corresponding threshold (too many users, not enough swap space, etc.), then the host is regarded as busy, and LIM will not recommend jobs to that host.

The CPU run queue length threshold values (`r15s`, `r1m`, and `r15m`) are taken as effective queue lengths as reported by `egosh resource view`.

All of these fields are optional; you only need to configure thresholds for load indices that you wish to use for determining whether hosts are busy. Fields that are not configured are not considered when determining host status. The keywords for the threshold fields are not case sensitive.

Thresholds can be set for any of the following:

- The built-inSymphony load indexes (r15s, r1m, r15m, ut, pg, i t, i o, l s, swp, mem, tmp)
- External load indexes defined in the Resource section of ego. shared

ResourceMap section

The ResourceMap section defines shared resources in your cluster. This section specifies the mapping between shared resources and their sharing hosts. When you define resources in the Resources section of ego. shared, there is no distinction between a shared and non-shared resource. By default, all resources are not shared and are local to each host. By defining the ResourceMap section, you can define resources that are shared by all hosts in the cluster or define resources that are shared by only some of the hosts in the cluster.

This section must appear after the Host section of ego. cluster. *cluster_name*, because it has a dependency on host names defined in the Host section.

ResourceMap section structure

The first line consists of the keywords RESOURCENAME and LOCATION. Subsequent lines describe the hosts that are associated with each configured resource.

Example ResourceMap section

```
Begin ResourceMap
RESOURCENAME    LOCATION
verilog         (5@[all])
local           ([host1 host2] [others])
End ResourceMap
```

The resource `verilog` must already be defined in the RESOURCE section of the ego. shared file. It is a static numeric resource shared by all hosts. The value for `verilog` is 5. The resource `local` is a numeric shared resource that contains two instances in the cluster. The first instance is shared by two machines, `host1` and `host2`. The second instance is shared by all other hosts.

LOCATION

Description

Defines the hosts that share the resource

For a static resource, you must define an initial value here as well. Do not define a value for a dynamic resource.

instance is a list of host names that share an instance of the resource. The reserved words `all`, `others`, and `default` can be specified for the instance:

`all` — Indicates that there is only one instance of the resource in the whole cluster and that this resource is shared by all of the hosts

Use the not operator (~) to exclude hosts from the `all` specification. For example:

```
(2@[all ~host3 ~host4])
```


means that 2 units of the resource are shared by all server hosts in the cluster made up of host 1 host2 . . . host*n*, except for host3 and host4. This is useful if you have a large cluster but only want to exclude a few hosts.

The parentheses are required in the specification. The not operator can only be used with the `all` keyword. It is not valid with the keywords `others` and `default`.

`others` — Indicates that the rest of the server hosts not explicitly listed in the `LOCATION` field comprise one instance of the resource

For example:

```
2@[host1] 4@[others]
```

indicates that there are 2 units of the resource on host 1 and 4 units of the resource shared by all other hosts.

`default` — Indicates an instance of a resource on each host in the cluster

This specifies a special case where the resource is in effect not shared and is local to every host.

`default` means at each host. Normally, you should not need to use `default`, because by default all resources are local to each host. You might want to use `ResourceMap` for a non-shared static resource if you need to specify different values for the resource on different hosts.

RESOURCENAME

Description

Name of the resource

This resource name must be defined in the `Resource` section of `ego.shared`. You must specify at least a name and description for the resource, using the keywords `RESOURCENAME` and `DESCRIPTION`.

- A resource name cannot begin with a number.
- A resource name cannot contain any of the following characters:

```
: . ( ) [ + - * / ! & | < > @ =
```

- A resource name cannot be any of the following reserved names:

```
cpu cpuf io logins ls idle maxmem maxswp maxtmp type model status it
```

```
mem ncpus define_ncpus_cores define_ncpus_procs
```

```
define_ncpus_threads ndisks pg r15m r15s r1m swap swp tmp ut
```

- To avoid conflict with `inf` and `nan` keywords in 3rd-party libraries, resource names should not begin with `inf` or `nan` (upper case or lower case). Resource requirement strings, such as `-R "infra"` or `-R "nano"` will cause an error. Use `-R "defined(infix)"` or `-R "defined(nanxx)"`, to specify these resource names.
- Resource names are case sensitive
- Resource names can be up to 39 characters in length

Environment Variables

Symphony client environment variables

This topic describes the environment variables that the Symphony client uses.

API_CALL_TIMEOUT

Syntax

```
API_CALL_TIMEOUT=value
```

Description

Defines the number of seconds the Symphony API waits to complete a connection to the session director or session manager before it times out.

For a client to submit workload, the client must connect to an application and then interact with a session created on this connection.

The timeout value can be any unsigned integer and -1, which is treated specially to specify that the API never times out.

Do not specify zero (0) because then the API never waits and so never connects.

Default

-1 (never times out)

RS_REQUEST_TIMEOUT

Syntax

```
RS_REQUEST_TIMEOUT=value
```

Description

Specifies the number of seconds the deployment command waits for the Repository Server (RS) to send or receive a service package to or from the RS before it times out.

The timeout value can be any positive integer and -1, which specifies that the deployment command never times out.

If you specify zero (0), the default value of 300 seconds is applied.

Default

300 (seconds)

SOAM_ENABLE_CLIENT_METADATA

Syntax

```
SOAM_ENABLE_CLIENT_METADATA=true | false
```

Description

Specifies whether the SSM collects metadata for the following attributes:

- client host name
- client IP address
- client OS user name

When `SOAM_ENABLE_CLIENT_METADATA` is set to false, the attributes are not collected or displayed. Use this environment variable to disable metadata collection to gain a performance boost in the session run time.

Default

true

SOAM_DIRECT_DATA_PORT

Syntax

```
SOAM_DIRECT_DATA_PORT=port_number / port_number_range
```

Description

Defines a port or port range for the client to listen for connections from the service; e.g., `SOAM_DIRECT_DATA_PORT="25000"` or `SOAM_DIRECT_DATA_PORT="25000-25100"`. You may want to do this if your client is running behind a firewall.

Default

Not applicable

Client reconnection environment variables

The following are the four environment variables that you can define on your client machine to influence how the client behaves when it loses its connection to the session manager:

- `SOAM_RECONNECTION_RETRY_INTERVAL`
- `SOAM_RECONNECTION_RETRY_LIMIT`
- `SOAM_RELOCATED_RECONNECTION_RETRY_INTERVAL`
- `SOAM_RELOCATED_RECONNECTION_RETRY_LIMIT`

How variables work together

When a client loses its connection to the session manager, it tries to reconnect to the existing session manager up to the `SOAM_RECONNECTION_RETRY_LIMIT`. Each reconnection attempt is `SOAM_RECONNECTION_RETRY_INTERVAL` seconds apart.

- If the client succeeds in reconnecting to a session manager, the API attempts to refresh all sessions on the connection. If the session manager was relocated during the disconnection, any non-recoverable sessions are aborted.
- If reconnection was unsuccessful:

The API stops trying to reconnect to the existing session manager and attempts to locate a new session manager. It then tries to locate and connect to a new session manager up to the `SOAM_RELOCATED_RECONNECTION_RETRY_LIMIT`. Each reconnection attempt is `SOAM_RELOCATED_RECONNECTION_RETRY_INTERVAL` seconds apart.

- If the client fails to reconnect to a session manager, the API stops trying to reconnect and throws an exception indicating that Symphony is no longer available.

- If the client succeeds in reconnecting to a session manager, the API attempts to refresh all sessions on the connection. If the session manager was relocated during the disconnection, any non-recoverable sessions are aborted.

Important:

Compatibility Note: The variables `CONNECTION_RETRY_INTERVAL` and `CONNECTION_RETRY_LIMIT` have been deprecated as of Symphony 4.0. This means that they continue to work in Symphony 5.1 but you are encouraged to use the variables that replace them. If you use the deprecated variables, all previous variable values are preserved. If you use any of the new variables, any definition of the deprecated variables is ignored and all new variable defaults are used. Mixing the deprecated variables with the newer variables always results in the newer variables and their default values (if not specified) being applied.

CONNECTION_RETRY_INTERVAL

Syntax

```
CONNECTION_RETRY_INTERVAL=value
```

Description

This variable is deprecated as of Symphony 4.0 and is replaced by `SOAM_RECONNECTION_RETRY_INTERVAL` and `SOAM_RELOCATED_RECONNECTION_RETRY_INTERVAL`. If you used this variable with previous versions of Symphony, it still works the way it used to.

If you need to influence the tolerance of a broken connection between your client and the system, use the following variables:

- `SOAM_RECONNECTION_RETRY_INTERVAL`
- `SOAM_RECONNECTION_RETRY_LIMIT`
- `SOAM_RELOCATED_RECONNECTION_RETRY_INTERVAL`
- `SOAM_RELOCATED_RECONNECTION_RETRY_LIMIT`

Default

3 (seconds)

CONNECTION_RETRY_LIMIT

Syntax

```
CONNECTION_RETRY_LIMIT=value
```

Description

This variable has been deprecated as of Symphony 4.0 and has been replaced by `SOAM_RECONNECTION_RETRY_LIMIT` and `SOAM_RELOCATED_RECONNECTION_RETRY_LIMIT`. If you have used this variable with previous versions of Symphony it will still work in the way it used to.

If you need to influence the tolerance of a broken connection between your client and the system, use the following variables:

- SOAM_RECONNECTION_RETRY_INTERVAL
- SOAM_RECONNECTION_RETRY_LIMIT
- SOAM_RELOCATED_RECONNECTION_RETRY_INTERVAL
- SOAM_RELOCATED_RECONNECTION_RETRY_LIMIT

Default

5 (times)

SOAM_RECONNECTION_RETRY_INTERVAL

Syntax

```
SOAM_RECONNECTION_RETRY_INTERVAL=value
```

Description

Specifies how long the API waits, in seconds, before the next attempt to reconnect to the last known instance of the session manager. This value is used in conjunction with SOAM_RECONNECTION_RETRY_LIMIT.

Note:

On Windows, the effective minimum interval is 1 second, even if you set the value to 0.

Default

1 (second)

SOAM_RECONNECTION_RETRY_LIMIT

Syntax

```
SOAM_RECONNECTION_RETRY_LIMIT=value
```

Description

Specifies the number of attempts made by the API to reconnect to the last known instance of the session manager. This variable is used to handle the case of a transient disconnection from the system for any type of session. When a disconnection is sensed by the API, instead of throwing an exception for a pending operation, the API attempts to automatically reconnect to the last known instance of the session manager for the specified number of times. When the API has reached the number of attempts without reconnection, it attempts to connect to a new instance of the session manager and uses SOAM_RELOCATED_RECONNECTION_RETRY_LIMIT.

Default

15 (times)

SOAM_RELOCATED_RECONNECTION_RETRY_LIMIT

Syntax

```
SOAM_RELOCATED_RECONNECTION_RETRY_LIMIT=value
```


Description

Specifies the number of attempts made by the API to reconnect to a new instance of the session manager.

This variable is used to handle the case in which session manager is relocated (either restarted on the same host or failed over to a different host). This variable is used by the API to locate a new instances of the session manager after SOAM_RECONNECTION_RETRY_LIMIT is reached.

Note:

On Windows, the effective minimum interval is 1 second, even if you set the value to 0.

Default

60 (times)

SOAM_RELOCATED_RECONNECTION_RETRY_INTERVAL

Syntax

```
SOAM_RELOCATED_RECONNECTION_RETRY_INTERVAL=value
```

Description

Specifies how long the API waits, in seconds, before the next attempt to reconnect to the new instance of the session manager. This value is used in conjunction with SOAM_RELOCATED_RECONNECTION_RETRY_LIMIT.

Default

5 (seconds)

Command line environment variables

SOAM_CLI_VIEWSESSION_COMPATIBILITY

Syntax

```
SOAM_CLI_VIEWSESSION_COMPATIBILITY=ON | OFF
```

Description

As of Symphony 5.1, the number of SIMs supported for one application can be up to 10000. However, in the output of the "soamview session application_name" command, the "RUN" column and "INST" column cannot support the display of such a large number. For example, 10000 will be displayed as "100*" in the "RUN" and "INST" columns of the command output. Therefore, to accommodate the maximum number of SIMs supported (10000), the output format has changed. To keep the total length of one line consistent with previous versions, the "ERR" and "CANL" columns have each been reduced by 1 character in width.

To keep backward compatibility, this environment variable defines whether the "soamview session application_name" command displays output in the previous format (prior to Symphony 5.1) or in the new format. If the environment variable is set to "ON" (case insensitive), the output will be displayed in the previous format; otherwise, the output will be displayed in the new format.

Default

OFF

TCP connection environment variables

You can set the following environment variables on your client machine to define the characteristics of the TCP connection:

- PLATCOMMDRV_TCP_KEEPALIVE_TIME
- PLATCOMMDRV_TCP_NODELAY
- PLATCOMMDRV_TCP_RECV_BUFFER_SIZE
- PLATCOMMDRV_TCP_SEND_BUFFER_SIZE

PLATCOMMDRV_TCP_KEEPALIVE_TIME

Syntax

```
PLATCOMMDRV_TCP_KEEPALIVE_TIME=seconds
```

Description

Specifies the maximum time in seconds the system can take to detect a broken connection to and from Symphony processes originating from the shell where the variable is defined.

Specify a numeric value of 180 seconds (3 minutes) or more. Values lower than 300 seconds (5 minutes) are not recommended.

Restriction:

This variable is ignored for installations on Solaris, which does not support this TCP/IP setting on a per-socket basis.

Default

Connections by default use the operating system-dependable, system-wide TCP keep alive setting, which for most operating systems, is 7200 seconds, or 2 hours.

PLATCOMMDRV_TCP_NODELAY

Syntax

```
PLATCOMMDRV_TCP_NODELAY=1 | 0
```

Description

Enables or disables the Nagle algorithm for TCP sockets.

Specify 1 to enable, 0 to disable.

Default

1 (enabled).

PLATCOMMDRV_TCP_RECV_BUFFER_SIZE

Syntax

```
PLATCOMMDRV_TCP_RECV_BUFFER_SIZE=bytes
```


Description

Specifies the size for the TCP receive buffer for each Symphony connection.
Specify a numeric value greater than or equal to 65535 bytes (64KB).

Default

65535 (64KB).

PLATCOMMDRV_TCP_SEND_BUFFER_SIZE

Syntax

```
PLATCOMMDRV_TCP_SEND_BUFFER_SIZE=bytes
```

Description

Specifies the size for the TCP send buffer for each Symphony connection.
Specify a numeric value greater than or equal to 65535 bytes (64KB).

Default

65535 (64KB).

System-defined environment variables

Environment variable	Description
SOAM_BINDIR	Directory where the Symphony middleware commands are installed
SOAM_HOME	Directory where the Symphony middleware is installed
SOAM_SERVER_DIR	Directory where the Symphony middleware daemons are installed

Session Director environment variables

This topic describes the environment variables that the Session Director (SD) uses in the `sd.xml` service definition file.

SD_RECOVERY_TIME

Syntax

```
<ego:EnvironmentVariable name="SD_RECOVERY_TIME">value</ego:EnvironmentVariable>
```

Description

Determines how long, in minutes, EGO will maintain the client allocation/activity, i.e., maintain in zombie status, while SD is down. If SD restarts during the specified period, it's allocation/activity is not lost. If SD does not restart during the specified period, its allocations will be lost as all SSM processes will be restarted.

Default

10 years

SOAM_SESSION_NAME_COMPATIBILITY

Syntax

```
<ego:EnvironmentVariable name="SOAM_SESSION_NAME_COMPATIBILITY">ON</ego:EnvironmentVariable>
```

Description

Determines if the valid character set for the session name is limited or not. If set to ON, there is no limitation, i.e., any characters can be used for the session name. If set to OFF, only alphanumeric, '-', and '_' characters are allowed.

Default

OFF

SOAM_OVERRIDE_APP_VERSION_WITH

Syntax

```
<ego:EnvironmentVariable name="SOAM_OVERRIDE_APP_VERSION_WITH">5.1</ego:EnvironmentVariable>
```

Description

Allows the middleware versions of all applications in the cluster to be overridden without any change to the application profiles.

Default

Not applicable

EGO environment variables

Environment variables are used to set the environment for commands, daemons and processes.

On UNIX, the EGO environment variables are set by the script `profile.ego` or `csSRC.ego`.

On Windows, the EGO environment variables are set by the installer.

Environment variables are primarily used internally by the software, but can be used as shortcuts to locate a particular directory.

Environment variable	Description	Default Value
EGO_BINDIR	The directory where commands are installed. Added to PATH on Linux and Path on Windows.	On Linux: <i>EGO_TOP</i> /1.2/platform/bin On Windows: <i>EGO_TOP</i> \1.2\bin
EGO_CLIENT_ADDR	Used with firewalls and rfa. Range of listening ports for clients to connect back to rfa. This environment is useful if you have a firewall configured on the client side. Example: EGO_CLIENT_ADDR=56000-56020	There is no default value. If not set, the client will listen on a random port.
EGO_CONFDIR	The directory where the valid EGO configuration file <code>ego.conf</code> is stored. The file may be duplicated in the system, but the cluster only uses the file stored in this location.	On Linux: <i>EGO_TOP</i> /kernel/conf or <i>\shared_dir</i> /kernel/conf On Windows: <i>EGO_TOP</i> \kernel\conf or <i>\shared_dir</i> \kernel\conf
EGO_ESRVDIR	The directory where EGO service configuration files are stored.	On Linux: <i>EGO_TOP</i> /eservice or <i>shared_dir</i> /eservice On Windows: <i>EGO_TOP</i> \eservice or <i>\shared_dir</i> \eservice
EGO_LIBDIR	The directory where the EGO libraries are installed, added to LD_LIBRARY_PATH on Linux, added to Path on Windows.	On Linux: <i>EGO_TOP</i> /1.2/platform/lib On Windows: <i>EGO_TOP</i> \1.2\lib

Environment variable	Description	Default Value
EGO_LOCAL_CONFIGDIR	This is the local configuration directory.	On Linux: <i>EGO_TOP/kernel /conf</i> On Windows: <i>EGO_TOP\kernel \conf</i>
EGO_SERVERDIR	The directory where the EGO server binaries and shell scripts are installed, added to PATH on Linux and Path on Windows.	On Linux: <i>EGO_TOP/1. 2/platform/etc</i> On Windows: <i>EGO_TOP\1. 2\etc</i>

Note:

\$EGO_TOP is the directory where EGO is installed, and *platform* represents the operating system. For example, for Linux: *linux2. 4-glibc2. 3-x86*

The most important environment variable to be set is EGO_CONFIGDIR, which, if not set in the current logon session, may prevent a user from running EGO clients.

Index

A

- aborted session state 110
- abortSessionIfClientDisconnect in application profile 193
- abortSessionIfTaskFail in application profile 194
- activities
 - viewing 25
- activity view subcommand 25
- administrators
 - removing 47
- ADMINISTRATORS
 - lsf.cluster file 281
- alloc list subcommand 25
- alloc modify subcommand 26
- alloc new subcommand 26
- alloc release subcommand 27
- alloc unblock subcommand 28
- alloc view subcommand 28
- allocation free subcommand 25
- allocations
 - displaying for client 25
 - displaying for consumer 25
 - reducing number of slots 27
 - releasing 30
 - removing for client 25
 - removing for consumer 25
 - requesting more slots 26
 - requesting new 26
 - unblocking a host 28
 - viewing 28
- API methods
 - defining time out 289
- API_CALL_TIMEOUT environment variable 289
- app -c in soamview 105
- app disable in soamcontrol 65
- app enable in soamcontrol 65
- app in soamview 104
- application profile
 - deploymentTimeout 201
 - how it is removed 101
 - updating 90
 - viewing 103
- application profile attributes
 - abortSessionIfClientDisconnect 193
 - abortSessionIfTaskFail 194
 - applicationName 137
 - consumerId 137
 - defaultResourceAttributeValue 198
 - delaySlotRelease 151, 203
 - description 199, 201–203
 - discardResultsOnDelivery 195
 - diskSpace 172, 174, 176, 177, 179, 183
 - elementName 163
 - enableCommonDataOptimization 197
 - enableSelectiveReclaim 152
 - enableStandbyServices 149
 - fileSwitchSize 168, 169
 - fileSwitchTime 169
 - flushDataAsap 150
 - ioRetryDelay 151
 - lastingPeriod 170, 204, 205
 - name (event) 164
 - name (Service) 199
 - numOfPreloadedServices 148
 - numOfSlotsForPreloadedServices 149
 - path 171, 173
 - persistSessionHistory 195
 - persistTaskHistory 194, 195
 - policy 142
 - pollFrequency 168
 - preemptionCriteria 152
 - preemptionRank 197
 - preemptionScope 153
 - preExecCmd 207
 - preference 197
 - preStartApplication 148
 - reclaimGracePeriod 192
 - recoverable 189

- resourceBalanceInterval 146
- resourceGroupFilter 196
- resourceGroupName 138, 160
- resReq 138, 156
- schedulingAffinity 153
- serviceToSlotRatio 196
- sessionSchedulingInterval 86, 147
- shutDownTimeout 156
- slotsThresholdForStandbyServices 154
- suspendGracePeriod 192
- taskCleanupPeriod 193
- taskGracePeriod 191
- taskLowWaterMark 146
- taskRetryLimit 190
- transientDisconnectionTimeout 150, 193
- value 165, 166
- applicationName in application profile 137
- applications
 - controlling 64
 - disabling 65
 - effects of unregistering an application 101
 - enabling 65
 - logging session manager messages 80
 - logging workload messages 78
 - naming 137
 - prioritizing sessions 86
 - registering 90
 - releasing resources 101
 - saving historic data 101
 - starting soamview counters 103
 - states
 - disabled and enabled 105
 - unregistering 101
 - updating 90
 - updating application profile 91
 - viewing disabled applications 105
 - viewing enabled applications 105

ARCHITECTURE

- Isf.shared file 273

B

- BEV_CRITICAL boundary event 165
- BEV_HALT boundary event 165
- BEV_PROACTIVE boundary event 165
- BEV_SEVERE boundary event 165
- blockHostOnTimeout parameter, application profile 167
- boundary events 163

- monitoring 165
- monitoring memory usage 165
- system responses 164
- triggering system responses 163
- boundary manager
 - monitoring physical memory 163
 - monitoring virtual address space 163
 - monitoring virtual memory 163

C

- client list subcommand 29
- client reg subcommand 29
- client unreg subcommand 30
- client view subcommand 30
- clients
 - defining behavior when disconnected 193
 - decreasing response time 148
 - generating list of 29
 - interaction with EGO 238
 - optimizing performance 148
 - problem running
 - EGO_CONFDIR environment variable 298
 - recovering from lost connections 289
 - registering 29
 - unregistering 30
 - viewing information about 30
- closed session state 110
- cluster
 - name
 - displaying 36
 - restarting 36
 - starting 37
 - starting all components 50
 - stopping 37
 - stopping all components 49
- cluster administrators
 - listing 47
- ClusterName
 - Isf.shared file 272
- clusters
 - changing priority of all sessions 86
 - defining hosts 264
- command line
 - license info 38
 - logging off 46
 - logging on 46
 - quitting 38

commands

- egoconfig 11
- egoconfig join 11
- egoconfig masterlist 11
 - egoconfig mghost
 - on UNIX 11
 - on Windows 12
 - egoconfig mghost soam
 - on UNIX 12
 - on Windows 13
- egoconfig unsetmghost 14
- egoremoverc 15
- egostrc 16
- egostrsudoers 17
- egosh activity view 25
- egosh alloc list 25
- egosh alloc modify 26
- egosh alloc new 26
- egosh alloc release 27
- egosh alloc unblock 28
- egosh alloc view 28
- egosh allocation free 25
- egosh client list 29
- egosh client reg 29
- egosh client unreg 30
- egosh client view 30
- egosh consumer alloc 30
- egosh consumer list 31
- egosh consumer view 31
- egosh debug pemoff 31, 32
- egosh debug pemon 31, 32
- egosh debug vemkdoff 33
- egosh debug vemkdon 33
- egosh ego elimrestart 35
- egosh ego info 36
- egosh ego restart 36
- egosh ego shutdown 37
- egosh ego start 37
- egosh exit 38
- egosh license info 38
- egosh quit 38
- egosh resource close 38
- egosh resource group 39
- egosh resource list 40
- egosh resource list -R 41
- egosh resource open 42
- egosh resource removehost 42
- egosh service list 43
- egosh service start 43
- egosh service stop 43
- egosh service view 43
- egosh user add 44
- egosh user assignrole 44
- egosh user delete 45
- egosh user execpasswd 34
- egosh user list 45
- egosh user logoff 46
- egosh user logon 46
- egosh user modify 46
- egosh user roles4user 47
- egosh user unassignrole 47
- egosh user users4role 47
- egosh user view 48
- egoshutdown 49
- egostartup 50
- prestartApplication 155
- pversions 51
- rsdeploy 52, 58
- running in a batch file 207
- running in a script 207
 - soamcontrol
 - app disable subcommand 65
 - app enable subcommand 65
 - session kill subcommand 66, 67
 - session resume subcommand 70, 71
 - session suspend subcommand 68, 69
 - soamdeploy
 - remove subcommand 75, 99
 - view subcommand 76
 - soamlog
 - sd subcommand 79
 - sim subcommand 81, 82
 - ssm subcommand 80
 - workload subcommand 83
- soamlogoff 84
- soamlogon 85
- soammod session 86
- soamreg 90
- soamshutdown 94
- soamstartup 95
- soamunreg 101, 119
 - soamview
 - app subcommand 104, 105
 - session subcommand 107–109, 112
 - task subcommand 113, 116
- specifying

- session password 85
 - session user name 85
- common data
 - optimizing paging 179, 181, 182, 184
 - specifying blocksize 179, 181, 182, 184
- common data updates 164
- compute host
 - configuring number of CPU slots 265
- compute hosts
 - allocating for consumer 138
 - allocating for session manager 156
 - defining 264
 - deploying packages 73
 - processes 94
 - shutting down 94
 - starting up 95
- compute resources
 - CPU slots 265
- configuration
 - adding local host to cluster 11
 - automatic startup
 - removing 15
 - configuration master candidates list 11
 - demote management host to compute host 14
 - host members 264
 - locating files 264
 - move to shared configuration file from local 11–13
 - root privileges 17
- configuration files
 - soam.conf 256
 - vem_resource.conf 95, 264
- consumer administrators
 - listing 47
- consumer list subcommand 31
- consumer users
 - listing 47
- consumer view subcommand 31
- consumerId in application profile 137
- consumers
 - allocation and demand information, summary 30
 - listing for a client 31
 - listing for a host 31
 - listing for allocation ID 31
 - summary of allocation and demand information 30
 - viewing information about 31
- control
 - controlling tasks, sessions, and applications 64
- cores

- setting cluster to 258
- counters
 - archive period 170
 - polling frequency 168
 - session manager shutdown period 156
 - switching files 169
 - task grace period 191, 192
 - transientDisconnectionTimeout 150
- CPU
 - factors
 - static resource 140, 158, 265
 - utilization
 - ut load index 243
- CPU factor (cpuf) static resource 140, 158
- CPU slots
 - configuring 265
- CPU_factor in vem_resource.conf 265
- cpuf static resource 140, 158, 265
- CPUFACITOR
 - lsf.shared file 274
- CPUs
 - specifying 265

D

- daemons
 - egosc 227
 - lim 229
 - pem 233
 - vemkd 237
- data 163
 - archived history files 170
 - common data
 - scheduling policy for data affinity 143
 - history files 170
 - persisting task history 194
 - specifying maximum age of history file 169
 - specifying maximum history file size 168
 - triggering file archival 168
 - tuning
 - monitoring available memory 163
 - triggering memory low conditions 163
 - triggering memory low events 165
- data files
 - specifying path
 - history 175, 178, 182
 - journaling sessions 175, 178, 182
 - journaling tasks 175, 178, 182

- paging common data 175, 178, 182
 - paging tasks 175, 178, 182
 - storing 175, 178, 182
- debug
 - of pem
 - turning off 31, 32
 - turning on 31, 32
 - of vemkd
 - turning off 33
 - turning on 33
- debug information
 - service instance manager 81, 82
 - workload logging levels 79–81, 83
- debug log levels
 - workload 79–81, 83
- debug pemoff subcommand 31, 32
- debug pemon subcommand 31, 32
- debug vemkdoff subcommand 33
- debug vemkdon subcommand 33
- dedicated resource. *See* exclusive resource 284
- delaySlotRelease in application profile 151, 203
- deploymentTimeout 201
- DESCRIPTION
 - Isf.shared file 276
- description in application profile 199, 201–203
- directories
 - for binaries 297
 - for EGO libraries 297
 - for EGO services 297
 - for ego.conf 297
 - for server binaries 298
- disabled application state 105
- disk space
 - defining maximum size 172, 174, 176, 177, 179, 183
 - for journaling 172, 174, 176, 177, 179, 183
 - for paging 172, 174, 176, 177, 179, 183
- disks
 - I/O rate 243
- diskSpace in application profile 172, 174, 176, 177, 179, 183

E

- effective run queue length 242
- ego info subcommand 36
- ego restart subcommand 36
- ego shutdown subcommand 37
- ego start subcommand 37
- EGO_BINDIR environment variable 297
- EGO_CONFDIR environment variable 297

- EGO_ESRVDIR environment variable 297
- EGO_HOST_ADDR_RANGE
 - ego.cluster file 278
- EGO_LIBDIR environment variable 297
- EGO_SERVERDIR environment variable 298
- EGO_STARTUP_ALTERNATE_PATHS
 - ego.sudoers file 268
- EGO_STARTUP_PATH
 - ego.sudoers file 268
- EGO_STARTUP_USERS
 - ego.sudoers file 269
- ego.conf
 - EGO_DEFINE_NCPUS 258
- ego.conf file
 - directory where stored 297
- ego.sudoers file 267
 - creating 17
- egoconfig command 11
- egoremovevc command 15
- egosetrc command 16
- egosetsudoers command 17
- egosh command
 - getting help 18
- egoshutdown command 49
- egostartup command 50
- elementName in application profile 163
- ELIM_POLL_INTERVAL
 - Isf.cluster file 277
- enabled application state 105
- enableSelectiveReclaim in application profile 152
- enableStandbyServices in application profile 149
- environment variables
 - API_CALL_TIMEOUT 289
 - defining path for preExec commands 207
 - EGO_BINDIR 297
 - EGO_CONFDIR 297
 - EGO_ESRVDIR 297
 - EGO_LIBDIR 297
 - EGO_SERVERDIR 298
 - list of 297
- errors
 - changing details logged 78
- events
 - boundaries 165
 - memory boundaries 163
 - naming 163
- exclusive resource 284
- execution

- daemons 233
- execution priority 140, 158
- execution sessions
 - closing 119
 - create session, run command and closing session at once 119
 - creating 119
- EXINTERNAL 280
- exit subcommand 38

F

- failover
 - facilitating 175, 178, 182
 - recovering sessions 189
- files
 - determining when to archive 168
 - facilitating recovery 161
 - monitoring frequency 168
 - polling frequency 168
 - sd.log4j.properties 79
 - sim.log4j.properties 81, 82
 - SOAM_HOME/conf/vem_resource.conf 264
 - specifying location of working files 161
 - storing on shared file system 161
 - vem_resource.conf 264
 - working and temporary 161
- fileSwitchSize in application profile 168
- fileSwitchTime in application profile 169
- flushDataAsap in application profile 150
- free memory 243

G

- grace periods
 - overriding 94
 - specifying for suspended sessions 191, 192
 - specifying for task cleanup 193

H

- historic data
 - saving 101
- history files
 - archiving 168
 - deleting archived files 170
 - logging historical information 168
 - polling 168

- renaming 168
- retaining archived files 170
- specifying archive retention period 170
- specifying maximum size 168
- specifying retention time for archived files 170
- hname static resource 140, 158
- host model static resource 140, 158
- host models
 - automatic detection 274
- host name static resource 140, 158
- host scavenging subcommand
 - ego elimrestart 35
- host type static resource 140, 158
- host types
 - automatic detection 274
 - displaying list of 40
- host_name in vem_resource.conf 265
- HOSTNAME
 - lsf.cluster file 283
- hosts
 - adding local to cluster 11
 - changing log levels 79
 - closing 38
 - configuring master candidates list 11
 - defining 264
 - displaying groups of 39
 - exclusive resource 284
 - opening 42
 - removing 42
 - shutting down 94
 - shutting down a management host 94
 - shutting down all 94
 - shutting down local 94
 - starting up 95

I

- idle time
 - built-in load index 243
- INCREASING
 - lsf.shared file 276
- install_dir/conf/vem_resource.conf 264
- INTERVAL
 - lsf.shared file 276
- io load index 243
- ioRetryDelay in application profile 151
- IPv6
 - in LSF_HOST_ADDR_RANGE 279

- it load index
 - description 243

J

- join subcommand 11

L

- lastingPeriod in application profile 170, 204, 205

- license info subcommand 38

- list
 - of user accounts
 - displaying 45

- load average 242

- load indices

- io 243
 - it 243
 - ls 243
 - mem 243
 - pg 243
 - r15m 242
 - r15s 242
 - r1m 242
 - swp 243
 - tmp 243
 - ut 243

- load management

- daemons 229

- LOCATION

- lsf.cluster file 285

- log files

- effects of closing applications 78
 - effects of restarting 78

- log levels

- changing 78
 - effecting changes 78
 - workload 79–81, 83

- logging levels

- setting to LOG_TRACE 32, 33

- login sessions 243

- ls load index 243

- lsf.cluster file 277

- lsf.cluster_name.license.acct file 270

- lsf.shared file 272

M

- management hosts

- configuring 264

- processes

- rs 94

- sd 94

- ssm 94

- start_agent 94

- shutting down 94

- starting 95

- master candidates

- displaying list of 40

- master host

- daemons 237

- displaying name of 36

- masterlist subcommand 11

- max _SSMs in vem_resource.conf 265

- max _SIMs in vem_resource.conf 265

- maxmem static resource 140, 158

- maxswp static resource 140, 158

- maxtmp static resource 140, 158

- mem load index 243

- memory

- available 243

- memory low conditions

- causes of 163

- correcting 164

- messages

- changing details logged 78

- effects of message blocksize 164

- mghost soam subcommand 12, 13

- mghost subcommand 11, 12

- middleware packages

- deploymnet and removal 52, 58

- model

- lsf.cluster file 283

- model static resource 140, 158

- MODELNAME

- lsf.shared file 274

N

- name

- application profile attributes

- osTypes 203

- Service 199

- name (event) in application profile 164

- name (Service) in application profile 199

- ncpus static resource 140, 158

- ndisks static resource 140, 158

- normalized run queue length
 - description 242
- numOfPreloadedServices in application profile 148
- numOfSlotsForPreloadedServices in application profile 149

O

- open session state 110
- operating systems
 - naming relevant 203
- optimization
 - client response time 148
 - prestarting applications 148
- order string 141, 159, 247
- osType in vem_resource.conf 265

P

- packages
 - deploying 73
 - uncompressing 73
- paging block size 171, 173, 175, 177
- paging rate
 - description 243
 - load index 243
- passwords
 - changing 46
 - for Windows user account
 - setting on Linux 34
- patches
 - viewing installed on Windows 51
- path in application profile 171, 173
- pem
 - debugging
 - turning off 31, 32
- persistSessionHistory in application profile 195
- persistTaskHistory in application profile 195
- Platform EGO emulator 264
- policy in application profile 142
- pollFrequency in application profile 168
- port_number in vem_resource.conf 264
- ports for EGO emulator 264
- preemptionCriteria in application profile 152
- preemptionScope in application profile 153
- preExecCmd in application profile 207
- preStartApplication in application profile 148
- priority
 - modifying 86
- privileges
 - removing 47

- processes
 - logging service instance manager 78
 - logging session director 78
 - logging session manager 78
 - Platform EGO emulator 94, 264
 - prestarting 148
 - reasons for limiting on session manager 164
 - repository service 94
 - service instance 94
 - service instance manager 94
 - session director 94
 - session manager 94
 - shutting down 94
 - start_agent 94, 264
 - starting at workload request 95
 - starting on management hosts 95

- processors
 - setting cluster to 258

PRODUCTS

- lsf.cluster file 281
- profile
 - version of 136
- properties files
 - sd.log4j.properties 79–82
 - sim.log4j.properties 81, 82
 - ssm.log4j.properties 80
- pversions command 51

Q

- quit subcommand 38

R

- r15m load index
 - built-in resources 242
- r15s load index
 - built-in resources 242
- r1m load index
 - built-in resources 242
- reclaimGracePeriod in application profile 192
- recoverable in application profile 189
- recovery
 - defining lost connection behavior 289
 - defining path to journaling files 175, 178, 182
 - saving session information 189
 - transientDisconnectionTimeout 150
 - waiting for client reconnection 150

- remote jobs
 - execution priority 140, 158
- remove in soamdeploy 75, 99
- resource allocation
 - balancing 146
 - compute resources 138
 - consumer 138, 160
 - session manager 156
 - tuning
 - balance frequency 146
- resource close subcommand 38
- resource group subcommand 39
- resource groups
 - displaying information about 39
- resource list -R subcommand 41
- resource list subcommand 40
- resource open subcommand 42
- resource removehost subcommand 42
- resource requirement string 139, 157, 244
- resource requirements
 - operators 140, 158, 245
 - sorting 141, 159, 247
- resourceBalanceInterval in application profile 146
- resourceGroupName in application profile 138
- RESOURCENAME
 - lsf.cluster file 286
 - lsf.shared file 275
- resources
 - balancing 146
 - balancing interval 146
 - closing 38
 - CPU factor 265
 - CPU slots 265
 - displaying
 - information about 42
 - list that matches string 41
 - master candidates 40
 - displaying allocations for 25
 - displaying demand 39
 - displaying resource shares 39
 - freeing for client 25
 - freeing for consumer 25
 - grouping 139, 157, 244
 - monitoring boundary values 165
 - monitoring events 165
 - monitoring memory 163
 - opening 42
 - polling frequency 146
 - reallocating session compute resources 147
 - reducing request 27
 - releasing 27, 101, 146
 - removing 42
 - requesting 26, 146
 - requesting additional 26
 - scheduling sessions 147
 - sorting 141, 159, 247
 - specifying 139, 157, 244
 - viewing allocations of 28
- RESOURCES
 - lsf.cluster file 283
- rexpri static resource 140, 158
- roles
 - for users
 - assigning 44
- root permissions
 - granting 17
- rs process 94
- rsdeploy 58
- rsdeploy command 52, 58
- run queue
 - effective 242
 - normalized 242

S

- scheduling policies
 - R_MinimumServices 143
- scheduling policies, R_PriorityScheduling 144
- scheduling policies, R_Proportion 142
- schedulingAffinity in application profile 153
- schema
 - version of 136
- sd in soamlog 79
- sd process 94
- sd.log4j.properties 79–82
- security
 - automatically logging off 85
- selection strings
 - operators 140, 158, 245
- server static resource 140, 158
- service controller 227
- service execution
 - aborting sessions 193
 - custom service deployment 207
 - defining termination cleanup period 193
 - deploying a service 207

- killing a session 193
 - specifying reclaim grace period, tasks 192
 - specifying suspension grace period 191, 192
- service instance manager
 - changing log levels 78
 - prestarting 148
- service instance manager, max number of SIMs 265
- service instances
 - prestarting 148
- service list subcommand 43
- service start subcommand 43
- service stop subcommand 43
- service view subcommand 43
- serviceName 187
- services
 - copying to the local host 207
 - displaying information about 43
 - listing 43
 - specifying
 - name 199
 - starting 43, 227
 - stopping 43
 - viewing deployed service packages 76
- session director
 - changing log levels 78
- session in soamview 112
- session kill, soamcontrol subcommand 66
- session manager
 - changing log levels 78
 - limiting processes run by 164
 - locating executables 203
 - monitoring memory usage 163
 - prestarting 148
 - shutting down 156
 - specifying client reconnection period 150
 - specifying max for host 265
 - specifying shutdown timeout 156
- session resources
 - increasing and decreasing 86
- session resume in soamcontrol 70, 71
- session suspend in soamcontrol 68, 69
- session, in soammod 86, 88
- sessions
 - aborting 193
 - applying priority changes 86
 - changing priority 86
 - closing after task cleanup period 193
 - closing after task grace period 191, 192
 - controlling 64
 - default Symphony session length 85
 - defining when to abort 193
 - duration 85
 - expiring time to live 85
 - killing 67
 - modifying priority 86
 - overriding logon user 85
 - priority 86
 - priority number range 87, 88
 - reasons for terminating 164
 - recovering 189
 - resuming 70
 - retaining information 189
 - setting time to live 85
 - specifying highest priority 87, 88
 - specifying scheduling interval 147
 - specifying Symphony session length 85
 - states
 - aborted 110
 - closed 110
 - open 110
 - suspended 110
 - suspending 68, 69
 - task cleanup period 193
 - task grace period 191, 192
 - terminated by soamreg 91
 - terminated by soamunreg 101
 - terminating if task fails 193
 - terminating with soamcontrol 66, 67
 - viewing active sessions 109
 - viewing information 103
 - viewing running sessions 109
 - viewing suspended sessions 109
- sessionSchedulingInterval in application profile 147
- shutdownTimeout in application profile 156
- sim in soamlog 81, 82
- sim.log4j.properties 81, 82
- slots
 - releasing 27
 - specifying 265
- slotsThresholdForStandbyServices in application profile 154
- soamcontrol command 64
- soamdeploy command 73
- soamlog command 78
- soamlogoff command 84
- soamlogon command 85

- soammod command 86
- soamreg command 90
- soamshutdown command 94
- soamstartup command 95
- soamunreg command 101, 119
- soamview command 103
- ssm in soamlog 80
- ssm process 94
- ssm.log4j.properties 80
- start and end time
 - bhist 116
- start_agent in vem_resource.conf 264
- start_agent process
 - shutting down 94
 - specifying ports 264
 - starting up 95
- states
 - viewing canceled tasks 113
 - viewing done tasks 113
 - viewing pending tasks 113
 - viewing running tasks 113
- static resources
 - See also* individual resource names
 - description 139, 157
- suspended session state 110
- suspendGracePeriod in application profile 192
- swap space
 - load index 243
- swp load index
 - description 243

T

- task history
 - rerun 194
 - rerun counts 194
 - runtime 194
- task in soamview 113
- task, soamview subcommand 116
- taskGracePeriod in application profile 191
- taskLowWaterMark in application profile 146
- taskRetryLimit in application profile 190
- tasks
 - aborting after task cleanup period 193
 - accommodating large paging files 171, 173, 175, 177
 - canceling after task grace period 191, 192
 - clean up duration 193
 - continuing to run after suspension 191, 192

- pagging block size 171, 173, 175, 177
- retaining history 194
 - states
 - canceled 113
 - done 113
 - pending 113
 - running 113
 - suspending 191, 192
 - terminated with soamunreg 101
 - viewing by task state 113
 - viewing closed 113
 - viewing information 103, 116
- threads
 - setting cluster to 258
- time interval format
 - bhst 116
- time to live
 - specifying for a session 85
- tmp load index
 - description 243
- transientDisconnectionTimeout in application profile 150
- tuning
 - available memory 163
 - monitoring virtual address space 163
- type
 - lsf.cluster file 284
- TYPE
 - lsf.shared file 276
- type static resource 140, 158
- TYPENAME
 - lsf.shared file 273

U

- unsetmghost subcommand 14
- user accounts
 - assigning role to 44
 - changing 46
 - creating 44
 - deleting 45
 - listing 45
 - listing by role 47
 - listing roles for 47
 - removing role from 47
- user add subcommand 44
- user assignrole subcommand 44
- user delete subcommand 45
- user execpasswd subcommand 34
- user list subcommand 45

- user logoff subcommand 46
- user logon subcommand 46
- user modify subcommand 46
- user roles
 - assigning 44
 - listing by user 47
 - listing users 47
 - removing 47
- user roles4user subcommand 47
- user unassignrole subcommand 47
- user users4role subcommand 47
- user view subcommand 48
- users
 - assigning roles to 44
 - changing session priority 86
 - displaying information about 48
 - logging off 84
 - logging on 85
 - specifying password 85
 - specifying session length 85
- ut load index
 - built-in resource 243

V

- value in application profile 165, 166
- vem_resource.conf
 - configuration file 264
 - specifying attributes 265

- vem_resource.conf attributes
 - host_name 265
 - port_number 264
- vemkd
 - turning off debug 33
 - turning on debug 33
- version
 - displaying 36
 - of EGO
 - displaying 18
- view in soamdeploy 76
- virtual address space
 - monitoring 163
- virtual memory
 - defined 163
 - load index 243
 - monitoring 163

W

- Windows
 - user accounts
 - setting password for on Linux 34
- working files
 - storing on shared file system 175, 178, 182
- workload
 - changing log levels 78
- workloadapi in soamlog 83