Administering and Using EGO

Platform EGO
Version 1.2.3
January 2008

Platform™

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.

Contents

PART

I

Managing the Cluster

CHAPTER

1

At a Glance: Critical Concepts

What is EGO? An overview

Enterprise Grid Orchestrator™ (EGO) allows developers, administrators, and users to treat a collection of distributed software and hardware resources on a shared computing infrastructure (cluster) as parts of a single virtual computer.

EGO assesses the demands of competing business services (consumers) operating within a cluster and dynamically allocates resources so as to best meet a company's overriding business objectives. These objectives might include

- Reducing the time or the cost of providing key business services
- Maximizing the revenue generated by existing computing infrastructure
- Configuring, enforcing, and auditing service plans for multiple consumers
- Ensuring high availability and business continuity through disaster scenarios
- Simplifying IT management and reducing management costs
- Consolidating divergent and mixed computing resources into a single virtual infrastructure that can be shared transparently between many business users

EGO also provides a full suite of services to support and manage resource orchestration. These include cluster management, configuration and auditing of service-level plans, resource facilitation to provide fail-over if a master host goes down, monitoring and data distribution.

EGO is only sensitive to the resource requirements of business services; EGO has no knowledge of any run-time dynamic parameters that exist for them. This means that EGO does not interfere with how a business service chooses to use the resources it has been allocated.

How does EGO work? A general overview

Platform products work in various ways to match business service (consumer) demands for resources with an available supply of resources. While a specific application manager or consumer (for example, Platform Symphony or an LSF cluster) identifies what its resource demands are, EGO is responsible for supplying those resources. EGO determines the number of resources each consumer is entitled to, takes into account a consumer's priority and overall objectives, and then allocates the number of required resources (for example, the number of slots, virtual machines, or physical machines).

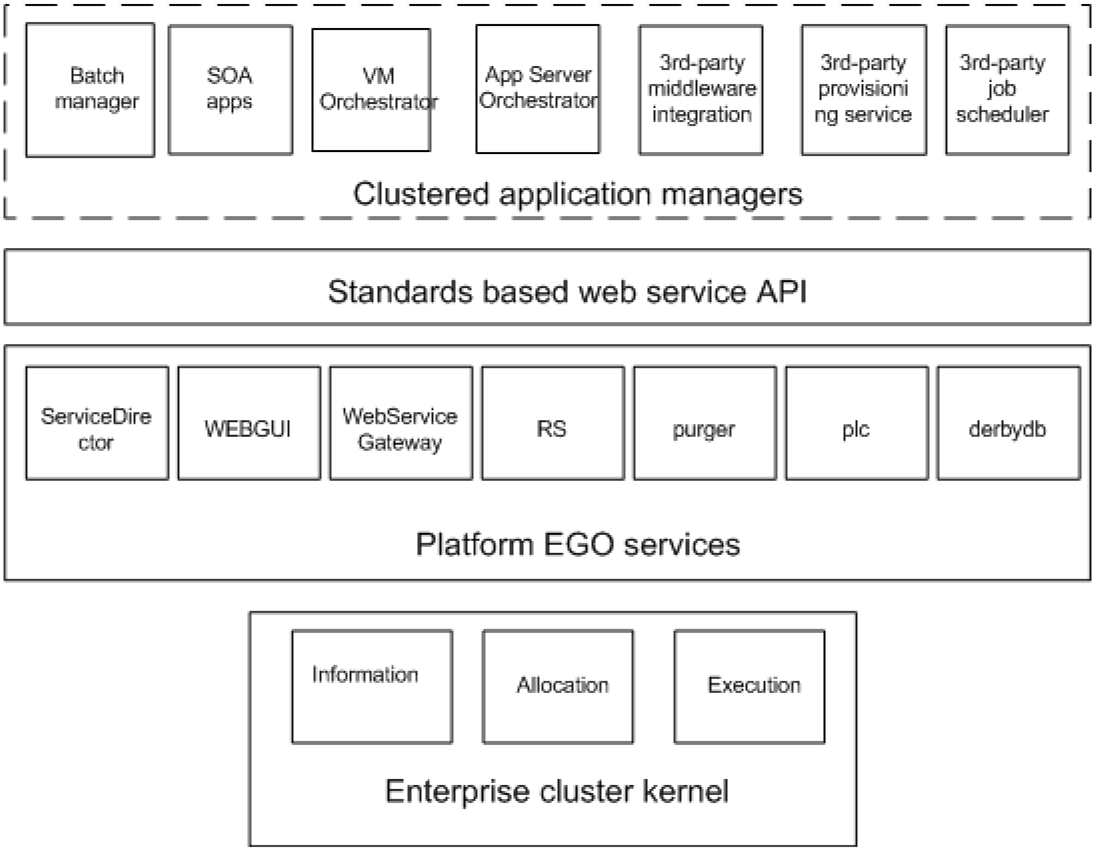
Once the consumer receives its allotted resources from EGO, the consumer applies its own rules and policies. How the consumer decides to balance its workload across the fixed resources allotted to it is not the responsibility of EGO.

So how does EGO know the demand? Administrators or developers use various EGO interfaces (such as the SDK or CLI) to tell EGO what constitutes a "demand" for more resources. When EGO identifies that there is a demand, it then distributes the required resources based on the resource plans given to it by the administrator or developer.

For all of this to happen smoothly, various components are built into EGO. Each EGO component performs a specific job. An illustrated overview of how these components fit within a larger system installation and interact with each other can be seen in the section EGO architecture. Definitions of each of the major components can be found in the section EGO components.

EGO architecture

EGO comprises a rich collection of cluster orchestration software components. The overall architecture is shown below:


Batch manager
SOA apps
VM Orchestrator
App Server Orchestrator
3rd-party middleware integration
3rd-party provisioning service
3rd-party job scheduler
Clustered application managers
Standards based web service API
ServiceDirector
WEBGUI
WebService Gateway
RS
purger
plc
derbydb
Platform EGO services
Information
Allocation
Execution
Enterprise cluster kernel

Important definitions

About…	Definition
Client	A piece of software that accesses system services through the API. The client acts as an agent for a consumer, making allocation requests on the consumer’s behalf.
Cluster	A cluster is a group of loosely coupled computers (hosts) that work together on a shared computing infrastructure. Within EGO, this group of hosts is managed by a single vemkd daemon that runs in the background on the master host. An EGO cluster has one master host that controls the rest of the cluster. Clients that interact with the cluster interact with the master host first, then are allocated cluster resources that they can use directly. The software automatically distributes the cluster resources fairly among various competing resource consumers. Once the cluster is constructed and running, an administrator defines consumers and users, adds resource distribution policies, deploys services, and registers applications that can run on the cluster. Then the cluster is ready to serve clients.
Consumers	A consumer represents an entity that can demand resources from the cluster. A consumer might be a business service, a business process that is a complex collection of business services, an individual user, or an entire line of business.
Host roles	Hosts in the cluster may be described as the master host, master candidates, management hosts, compute hosts, the web server host, or the DB host.
EGO_TOP and EGO_CONFDIR	This document uses EGO_TOP to describe the top-level EGO directory (where EGO is installed). %EGO_CONFDIR% (Windows) or $EGO_CONFDIR (Unix) is an environment variable and is used when the document is describing where configuration files can be found. When you have enabled failover, this directory differs from EGO_TOP.
Resources	Resources are physical and logical entities that can be requested by a client. For example, an application (client) requests a CPU (resource) to run. Resources also have attributes. For example, a host has attributes of memory, CPU utilization, operating systems type, etc.
Services	A system service is an internal service that is responsible for running part of the infrastructure. System services may have multiple concurrent service instances running on multiple hosts. All system services (except for derbydb) are automatically enabled by default at installation.

CHAPTER

2

Administrative Basics

Note:

Before continuing, be sure to install Platform EGO according to the best practices described in the various Platform EGO installation guides. A default installation is assumed.

Log on to the Platform Management Console

You have set your environment on this host. You are logged into the operating system as egoadmin.

The Platform Management Console allows you to monitor, administer, and configure your cluster.

1. If you do not already know the web server URL, run **egosh client view**.
   a) Look for the client name preceded by “GUIURL”.

      For example, `GUIURL_Host_W` (this is the fully qualified host name, which is the complete host and domain name "`Host_W.mycompany.com`").

   b) Look for the DESCRIPTION line beneath the client name to find the web server URL, and then copy it.

      For example: `http://Host_W:8080/platform`.
2. Launch any web browser and enter the address of the web server URL.

   The format of the URL is always `http://`*`host_name:port_number`*`/platform`
3. Log on to the Platform Management Console for the first time by specifying
   - User Name: Admin
   - Password: Admin

   For security in a production environment, we strongly recommend that you change the password of the Admin account.

Modify user password and other account information

You must log on as a valid user with the appropriate permissions to modify account information.

You can change user passwords from the Platform Management Console or the command line, but you can only modify account information from the command line.

1. To change your password from the Console:
   a) Go to the logon page and click Change Password.
   b) In the appropriate fields, enter your user name, current password, and a new password.
   c) Retype the new password to confirm it.
   d) Click Apply.

   The password can be made up of 1 to 16 alphanumeric or special characters, except control characters (Ctrl + key).
2. To change your password from the command line:
   a) Run **egosh user modify -u *user_account* -x *password***, specifying the name of the user account you wish to modify followed by its new password.

      The password can be made up of 1 to 16 alphanumeric or special characters, except control characters (Ctrl + key).

      Note:

      You cannot modify the name of the user account, only the password.

3. To modify other user account information from the command line, add any of the following optional commands to **egosh user modify -u *user_account***:

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

   - **-e** *email*: Specifies a new email address of the user to whom this account belongs.

     Specify up to 64 alphanumeric or special characters, except 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.

In Windows, after changing an Admin user password, the password must be changed for lim services on all management hosts. Be sure to run **egoconfig mghost** on each management host after making a password change.

Configure a secure Platform Management Console (https)

The `$JAVA_HOME` environment variable must be set correctly on the host where the WEBGUI service is running.

1. Stop the WEBGUI service:

   egosh service stop WEBGUI
2. Generate a self-signed certificate for internal use:
   a) Change directory to `$JAVA_HOME/bin`
   b) Type **./keytool -genkey -alias tomcat -keyalg RSA**

      A prompt for the keystore password displays.
   c) Use Tomcat's default password: **changeit**.
   d) When prompted for last and first name, enter the FQDN of the Platform Management Console host. For example:

      `sympmcserver.platform.com`
   e) Answer all other questions.
3. Enable the SSL.
   a) Find the SSL connector element in *`GUI_TOP`*`/tomcat/conf/server.xml`.

      The SSL connector element is located below this comment: `<!-- Define a SSL HTTP/1.1 Connector on port 8443 -->`
   b) Uncomment the SSL connector element by removing the comment tags: <!--and -->.
   c) Change the port number to 8443 in `$EGO_CONFDIR/../../gui/conf/wsm.conf`.
4. Start the WEBGUI service:

   egosh service start WEBGUI

The Platform Management Console is now accessible through https and through port 8443. For example: https://*WEBGUI_server*:8443/Platform

Understanding the dashboard

The dashboard is only visible to cluster and consumer administrators. If you are not logged in as a cluster or consumer administrator, you cannot see the dashboard.

Overview

The dashboard is a window into your cluster. Use the dashboard to get an overall picture of the health of your cluster and to get early warning for any systemic problems that may be occurring.

- Your hosts
- CPU utilization levels: What is right?

Note:

Your dashboard is dynamic and can change depending on what components you have integrated. For information about integration, contact Platform Support.

You can customize your tables on the dashboard and elsewhere by clicking Preferences.

Your hosts

The health of your hosts is one of the major elements of your cluster that can seriously affect its efficiency. Healthy hosts mean better throughput and a more stable production environment.

What you see

Your cluster view gives you a quick look at the health of your entire cluster at a glance. Use the Hosts by State and Hosts by CPU Utilization charts in the Cluster Health area to monitor how your hosts are doing and how hard they are working.

Beneath these charts, you can see the health of your master host and your master candidates if you have specified any. For your master host health, rest your mouse over the icon for a detailed description of its real-time attributes or click on the icon for detailed information about that particular host.

What to watch for

For your cluster to be healthy and working efficiently, your hosts have to be running or available to run workload units. They must also be able to run workload units at an acceptable CPU utilization level. Use the Platform Management Console to monitor for

- A large number of closed or unavailable hosts
- Hosts running at very high or very low CPU utilization

Either of these indicators means that your cluster is not running as efficiently as possible and may need some attention.

CPU utilization levels: What is right?

Only you can decide what the optimal CPU utilization level is for your workload. In some cases, 90% is acceptable and in others 70% is good. Try to recognize the level at which your hosts are attaining the highest CPU utilization level without becoming frequently unavailable.

Goal 1: Predictability and output

If you want your cluster to be as predictable as possible and for workload units to finish and produce results at a possible cost of speed, make sure your hosts are only running at 70% CPU utilization or less.

Goal 2: Fast turnaround time

If what you require is speed (you do not care if some workload units fail as long as most of them finish as quickly as possible), then 90% CPU utilization is probably the right level for you.

Understanding the host view

The host view in the Platform Management Console gives you all the information you need to keep an eye on your entire cluster and to quickly and easily spot those hosts that are experiencing any difficulties. Use Filter Settings to customize the view of your hosts. For example, you can filter hosts by state to display only unavailable or closed hosts.

You can also select Hosts (List View) and customize the metrics you see in the table with Preferences.

Healthy hosts mean that your workload is running optimally and you are making the best possible use of your resources. If your hosts are closed or unavailable, your workload suffers. The more hosts you have available to run workload units, the more robust your cluster becomes. The first step in keeping your cluster healthy is to monitor the health of your hosts.

For details about the performance of a specific host, rest your cursor on the icon (when using the Icon View). In either the Icon View or the List View, click on the icon or host name to get host properties.

Master host

Your master host is always displayed as an icon at the bottom of your host view. For details, rest your cursor on the icon. For master host properties, click the icon.

How can I tell my hosts are in trouble?

Your cluster only works as well as your hosts are performing. But how do you know when hosts are not working as well as possible? Use the hints below to get an idea of how you can use the Platform Management Console to monitor your host health.

Pattern

Use the icon view to display your hosts and their health visually. It is the pattern of the health of the hosts that is important to maintaining a healthy cluster. There should be few hosts Unavailable or Closed compared to OK hosts.

Height

The color height of the icons in the icon view gives you the CPU utilization of that host. Generally, in a healthy cluster, most hosts have between 7 and 9 bars in height, meaning they are using between 70% and 90% CPU utilization. Depending on the workload you are running, these thresholds may not be correct for you; they are general guidelines only.

Master host

Keep an eye on your master host at the bottom of the Host View. It should not be overloaded. Candidates are designed to back up your cluster should a master host become unavailable for any reason, but it is recommended that you safeguard the health of your master host as much as possible.

Host icons

Using the Platform Management Console’s icon host view, you can see the state of all your hosts.

Host icon	What it means
🟩🟩🟩	A green base with green bars means that your host is OK and that it is running workload units. The more CPU capacity a host is using, the higher the green bars. A green outline means that the host has been activated at the lowest priority (for host scavenging).
▤	A grey icon means that this host is closed. It may be closed for several reasons.
🟥	A red icon means that the host is unavailable.

The master host icon and name are displayed at the bottom of the Console page.

How do I see details?

For any host, hold your mouse over the icon for details about the current state of the host. For even more detail, click on the host icon. The Host Properties page opens with all the details for that host.

Turn off tooltips

Turning off Host (Icon View) tooltips in the Platform Management Console is helpful for performance tuning in clusters with more than 1000 hosts. If tooltips are enabled in clusters of this size, a 5-10 second delay may occur whenever you use the Host (Icon View) page.

1. With an XML editor, open `pmc_conf_ego.xml`.
   - Windows: *`EGO_TOP`*`\gui\conf\pmcconf\pmc_conf_ego.xml`
   - Linux: *`EGO_TOP`*`/gui/conf/pmcconf/pmc_conf_ego.xml`

   If you ran `egoconfig mghost`, then `pmc_conf_ego.xml` is located in the *EGOshare* directory:
   - Windows: *`EGOshare`*`\gui\conf\pmcconf\pmc_conf_ego.xml`
   - Linux: *`EGOshare`*`/gui/conf/pmcconf/pmc_conf_ego.xml`
2. In the configuration section, locate the parameter: <Name>hostTooltipsFlag</Name>.
3. In this parameter, change <Value>true</Value> to <Value>false</Value>.
4. Save and close the file.
5. Restart the WEBGUI service.

Tooltips for your Host (Icon View) page no longer appear. Only the host state displays when you roll the mouse over a host icon (for example, “OK”).

Set the command-line environment

On Linux hosts, set the environment before you run any commands. You need to do this once for each session you open. Both root and egoadmin accounts use EGO commands to configure and start the cluster.

You need to reset the environment if the environment changes during your session, for example, if you run `egoconfig mghost`, which changes the location of some configuration files.

- For `csh` or `tcsh`, use `cshrc.platform`.

  source EGO_TOP/cshrc.platform
- For `sh`, `ksh`, or `bash`, use `profile.platform`.

  . EGO_TOP/profile.platform

Parallel installation

You have installed a second cluster to run in parallel with an original cluster, but you have not started the second cluster yet.

If you have installed one cluster and want to install a second cluster in parallel with the first as a means of transitioning from the old cluster to the new cluster, you must make some changes after installing but before starting the second cluster.

1. Change the service start type from automatic to manual in `named.xml` located in `ego/eservice/esc/conf/services`.

   ```
   <sc:StartType>Manual</sc:StartType>
   ```

2. Comment out the dependency on Service Director in `wsg.xml` located in `ego/eservice/esc/conf/services`.

   ```
   <!-- <sc:Dependency type="OnStart">ServiceDirector</sc:Dependency>
   ```

3. Start the second cluster.
4. When the first cluster has been decomissioned, stop the WebServiceGateway service.
5. Change the start type back to manual in `named.xml` and remove the comment on service director dependency in `wsg.xml`.
6. Start the WebServiceGateway service.

Executing commands in a multi-cluster environment

Symphony users can run EGO commands from one host that may connect with one or more clusters and have this command take effect in a specific cluster.

Note:

All clusters must be the same version, for example, Symphony 3.2.

Applies only to Symphony users.

Install and set your environment to run egosh commands in a multi-cluster environment

Whether you are an interactive user or not, follow these steps to be able to run `egosh` commands in a multi-cluster environment.

You must complete these steps each time you want to be able to run `egosh` on a different cluster.

1. On the host that you want to run in a multi-cluster environment, install the Symphony client package.
2. Copy the folder `EGO_TOP\1.2\bin` from the master host to this host.
3. Set the environment variable PATH in the `symclientenv.bat` file under `SymphonyClient\conf` on the host to include the directory containing the `egosh` binary (`EGO_TOP\1.2\bin\egosh.exe`).
   a) If the PATH variable is not set, modify `symclientenv.bat` file so that it points %EGO_CONFDIR% of the Symphony shared directory of the cluster to be connected to. For example: `\\FileServer\SymShare\kernel\conf`.
   b) In `ego.conf` under `SymphonyClient\conf`, modify it by adding the master list and VEMKD port number.
4. Run `symclientenv.bat`.
5. Run `egosh resource list`.

   A list of resources displays if the configuration was successful.

You can now issue `egosh` subcommands from this host to another cluster.

Cluster management tasks (Windows)

- Start the Windows cluster
- Shut down the Windows cluster
- Restart the Windows cluster

Start EGO on the Windows cluster

You must have administrator access on the local Windows host.

1. If this is a Windows cluster, start EGO on all hosts in the cluster:

   egosh ego start all

   This starts all Windows hosts.
2. If this is a mixed cluster, start EGO on all Windows and all Linux hosts in this way:
   a) Log on to a Windows host in the cluster and run **egosh ego start all**.
   b) Log on to a Linux host in the cluster, set the environment variables, and then run **egosh ego start all**.

Shut down EGO on the Windows cluster

You must be logged on to any host in the cluster as the local system user and have EGO administrative privileges.

Note:

If the master is down, log on to a management host (if the master is down, there is no way to get the complete list of cluster services and daemons; on the management host, the cluster file and hostcache file lists this information).

---

1. Log on to EGO as a cluster administrator:

   egosh user logon -u *user_name* **-x** *password*

   For example, type

   egosh user logon -u Admin -x Admin
2. Shut down all EGO services and daemons on all Windows hosts in the cluster:

   egoshutdown.bat

Restart EGO on the Windows cluster

You must be logged on to any host in the cluster as the local system user and have EGO administrative privileges.

Note:

If the master is down, log on to a management host (if the master is down, there is no way to get the complete list of cluster services and daemons; on the management host, the cluster file and hostcache file lists this information).

---

1. Restart EGO.

   egosh ego restart all

EGO restarts on all hosts in the cluster.

Cluster management tasks (Linux)

- Start the Linux cluster
- Shut down the Linux cluster
- Restart the Linux cluster

Start EGO on the Linux cluster

Log on with root permissions. Ensure rsh is available on each host in the cluster.

Important:

By default, only root can start, stop, or restart the cluster. Optionally, you can grant root privileges to egoadmin, the cluster administrator account. See *Planning and Installing Your EGO Cluster on Linux* for details.

1. Start EGO on all Linux hosts in the cluster:

   egosh ego start all

Shut down EGO on the Linux cluster

Log on as egoadmin or root to any host in the cluster.

Note:

If the master is down, log on to a management host (if the master is down, there is no way to get the complete list of cluster services and daemons; on the management host, the cluster file and hostcache file lists this information).

1. Log on to EGO as a cluster administrator:

   egosh user logon -u *user_name* -x *password*

   For example, type

   egosh user logon -u Admin -x Admin
2. Shut down all EGO services and daemons on all Linux hosts in the cluster:

   egoshutdown.sh

Restart EGO on the Linux cluster

Log on with root permissions on the local host. Assuming the master is up, logon to any host in the cluster.

Note:

If the master is down, log on to a management host (if the master is down, there is no way to get the complete list of cluster services and daemons; on the management host, the cluster file and hostcache file lists this information).

1. Restart EGO:

   egosh ego restart all

EGO restarts on all hosts in the cluster (including Windows hosts if you have a mixed cluster). Services are not restarted with this command.

Grant root privileges to a cluster administrator

Optional. A root user within a Linux environment can choose to give root privileges within the cluster to the cluster administrator.

Check the following:

- That you are logged on as `root`.
- That `/etc/ego.sudoers` does not already exist. If the file does exist, use the -p option below.

By default, only root can start, stop, or restart the cluster.

Give root privileges to egoadmin so that egoadmin can start a local host in the cluster, or 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.

Do the following to give root privileges to egoadmin for one host. Run the command on each host in the cluster.

1. Run the `egosetsudoers.sh` command.

   Note:

   If you already have an `ego.sudoers` file from a previous cluster, run this command with the option -p.

   When you run `egosetsudoers.sh`, it does the following:

   It creates the `/etc/ego.sudoers` file. The file owner is `root` and the permissions are set to `600` because you ran this command as `root`. Only the root user can edit this file.

   It setuids the `egosh` command and change the owner of `egosh` to root.

   Whenever you see instructions to log on as root to start, stop, or restart a host in the cluster, you may log on as egoadmin instead.

View and modify cluster properties

1. Click Cluster > Summary.

   The Summary page displays.

2. Click Cluster Properties.
3. Modify the cluster properties.
   a) The cluster name cannot be changed after installation.
   b) Add or remove user accounts as cluster administrators.

      The default accounts are:

      - Admin: Set as cluster administrator by default.
      - Guest: Not set as cluster administrator by default.

   c) If desired, modify the resource allocation behavior for the cluster.

      - Reclaim shared resources: Reclaims resources from any consumer that has been allocated too many resources for its current needs.
      - Reclaim lent resources before borrowing: When there is an unmet allocation request, sets the cluster to reclaim resources that have been lent out before borrowing begins.

   d) Specify resource groups available to the entire cluster.

      By default, there are three resource groups created when you install and you cannot deselect these three because the system needs them for the cluster to function:

      - ComputeHosts: Once you have created your own resource groups, you can deselect this resource group for the cluster. By default, includes any host in your cluster that is not a management host. These hosts are available to run work.
      - InternalResourceGroup: Group of hosts that run internal processes.
      - ManagementHosts: Includes your master host and any failover candidates plus any other host where you have installed the management host package.

IPv6 support

EGO supports IPv6. Among other benefits of IPv6, longer address lengths (128 bit) result in increased address availability for networked devices, allowing you to allocate addresses in large blocks.

EGO and IPv6

Notes on EGO support for IPv6 include the following:

- Scope: You must assign global unicast addresses to IPv6-enabled hosts.
- Compatibility: IPv4 and IPv6 hosts can only communicate with corresponding hosts of the same type or with dual-stack hosts. However, dual-stack hosts can communicate with both IPv4 and IPv6 hosts, allowing compatibility between the two host types. Therefore, if you have a mixed cluster with both IPv4 and IPv6 hosts, configure it to be dual-stack.
- DNS cache: To locally store host names and mapped addresses (both IPv4 and IPv6 types), EGO uses a host DNS cache.
- Hosts file: You can define a `hosts` file to provide address-to-name translation. Create `hosts` in `$EGO_CONFDIR/kernel/conf` (Linux) or `%EGO_CONFDIR%\kernel\conf` (Windows). This file is most useful if you have multi-homed hosts (hosts with multiple interface cards/nics) or dual-stack hosts. It assists you in authenticating hosts, and viewing host mappings in cases where there are multiple addresses and names.

Configuring for IPv6 support

To enable, disable, and/or configure IPv6 support, set these parameters.

Parameter	**Variables**	**Default**	**File name and location**
EGO_DHCP_ENV	• Undefined (client's IP addresses are cached) • Defined (client's IP addresses are not cached; enables dynamic IP addressing for all client hosts in cluster) If defined, you must also define EGO_DYNAMIC_HOST_WAIT_TIME for hosts to rejoin a cluster after their IP address changes.	Undefined	• Linux: *`$EGO_CONFDIR/kernel/conf/ego.conf`* • Windows: `%EGO_CONFDIR%\kernel\conf\ego.conf`
EGO_DUALSTACK_PREFER_IPV6	• Y (returns IPv6 at front) • N (returns IPv4 at front) Meaningful for dual-stack hosts. If set, a dual-stack host uses IPv6 instead of IPv4 to communicate with other IPv6 or dual-stack hosts. (See note.)	N (IPv4 at front)	• Linux: `$EGO_CONFDIR/kernel/conf/ego.conf` • Windows: `%EGO_CONFDIR%\kernel\conf\ego.conf`

Parameter	Variables	Default	File name and location
EGO_ENABLE_SUPPORT_IPV6	• Y (to enable) • N (to disable) Enabling support for IPv6 does not have any effect IPv4-only hosts. If set to N, IPv6-only hosts are not recognized.	N (disabled)	• Linux: `$EGO_CONFDIR/kernel/conf/ego.conf` • Windows: `%EGO_CONFDIR%\kernel\conf\ego.conf`
EGO_HOST_CACHE_DISABLE	• Y (to disable) • N (to enable) Enabling support for IPv6 does not have any effect IPv4-only hosts.	N (enable caching of host names and addresses)	• Linux: `$EGO_CONFDIR/kernel/conf/ego.conf` • Windows: `%EGO_CONFDIR%\kernel\conf\ego.conf`
EGO_HOST_CACHE_NTTL	"Negative time to live", in seconds The amount of time that errors are cached.	20 seconds (to turn off caching completely, set to 0)	• Linux: *`$EGO_CONFDIR/`*`kernel/conf/ego.conf` • Windows: *`%EGO_CONFDIR%`*`\kernel\conf\ego.conf`
EGO_HOST_CACHE_PTTL	"Positive time to live", in seconds The amount of time cached results are stored.	86400 seconds (24 hours) (to turn off caching completely, set to 0)	• Linux: `$EGO_CONFDIR/kernel/conf/ego.conf` • Windows: `%EGO_CONFDIR%\kernel\conf\ego.conf`

Note:

Set EGO_DUALSTACK_PREFER_IPV6 in `ego.conf` to sort address presentation in the cache library. For example, set EGO_DUALSTACK_PREFER_IPV6 to “**Y**" so that IPv6 addresses appear at the top of the list; set it to “**N**" so that IPv4 addresses appear at the top. This parameter only affects the sort order of the list, and communication between dual-stack hosts. It does not enable IPv6, nor does it filter the address list.

Configuring IPv6 in multiple configuration files

If you have installed an application manager (for example, Platform LSF) on top of EGO, it is recommended to configure IPv6 support in `ego.conf`, not the `.conf` file belonging to the application manager (`lsf.conf`). Note the following interactions:

- If IPv6 is enabled in `ego.conf` and disabled in `lsf.conf`, Platform LSF uses IPv6.
- If IPv6 is disabled in `ego.conf` and enabled in `lsf.conf`, Platform LSF does not use IPv6.
- If IPv6 is not configured in `ego.conf` and enabled in `lsf.conf`, Platform LSF uses IPv6; however, some EGO daemons such as egosc do not use IPv6.

External user authentication and non-NFS file systems

The standard EGO package authenticates all users who log on to EGO using its built-in authentication. EGO can also integrate with external user authentication protocols for stronger and more sophisticated user account management, including the following:

- Kerberos™ 5 user authentication
- SiteMinder® user authentication
- OS user authentication, beneath EGO
- Any third-party user identity system

When using an external user authentication system, EGO users are authenticated using their identity in the external system.

EGO can also be installed on non-NFS file systems, such as AFS® and DCE/DFS™.

Definitions

- AFS: Andrew file system; a distributed networked file system
- DCE/DFS: distributed computing environment/distributed file system; a remote file access protocol used with the distributed computing environment
- Kerberos: An authentication protocol used to authenticate (user) identities in a secure manner across computer networks
- SiteMinder: A security identity management system for authentication and authorization; provides policy-based authentication and authorization, as well as Single Sign-On for all Web-based applications

Enable support for external authentication

To enable support for external authentication, you must replace the default security plug-in that comes with the standard EGO package with an external security plug-in. Contact support@platform.com to obtain the appropriate security plug-in.

Install EGO on non-NFS file systems (AFS, DCE/DFS)

To install EGO in an AFS or DCE/DFS environment, you require an installation package specific to your integration. Contact support@platform.com to obtain the appropriate integration package.

Transferring files from one host to another

Using the command line interface, you can transfer files between hosts. This is useful when installing or upgrading packages. Commands associated with data and file transfer include rfa list, rfa get, rfa put, and rfa remove.

- List host files
- Copy a file from another host
- Copy a file to another host
- Remove a file from a host

List host files

You must have appropriate privileges on each of the hosts involved. Directory path names must be absolute. You may wish to first request a resource allocation/slot from EGO before running the commands (if you do not, an allocation is automatically requested on your behalf).

Run the command `rfa list` to list file(s) from a local or remote directory.

1. From the command line, run the `rfa list` command.

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

   - **-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 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.

     Note:

     If you have a sequence of commands, there is no need log on or provide credentials each time. Note that credentials, which are locally stored, expire after a certain length of time.
   - **-u** *user_name*: (Optional.) Fully-qualified user name of the execution account to register the password for.
   - **-x** *password*: (Optional.) Password to register for the execution user account.

Note:

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 user_name and password.

Copy a file from another host

You must have appropriate privileges on each of the hosts involved in the file transfer. Directory path names must be absolute. You may wish to first request a resource allocation/slot from EGO before running the commands (if you do not, an allocation is automatically requested on your behalf).

Run the command `rfa get` to copy a file from a specified host to a local destination.

1. From the command line, run the `rfa get` command.

```
rfa get -t host_name -d local_file | -s remote_file -p consumer [-c
credential | -u user_name -x password]
```

- **-t** *host_name*: Name of host you want to copy a file from.
- **-d** *local_file*: Name of the local file, including its directory location, you want to copy to. Directory path must be absolute
- **-s** *remote_file*: Name of the remote file, including its directory location, you want to copy. Directory path must be absolute.
- **-p** *consumer*: 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.

  Note:

  If you have a sequence of commands, there is no need log on or provide credentials each time. Note that credentials, which are locally stored, expire after a certain length of time.
- **-u** *user_name*: (Optional.) Fully-qualified user name of the execution account to register the password for.
- **-x** *password*: (Optional.) Password to register for the execution user account.

Note:

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 user_name and password.

Copy a file to another host

You must have appropriate privileges on each of the hosts involved. Directory path names must be absolute. You may wish to first request a resource allocation/slot from EGO before running the commands (if you do not, an allocation is automatically requested on your behalf).

Run the command `rfa put` to copy a file to a specified host from a local location.

1. From the command line, run the `rfa put` command.

   ```
   rfa put -t host_name -d local_file | -s remote_file -p consumer [-c
   credential | -u user_name -x password]
   ```

   - **-t** *host_name*: Name of host you want to copy a file to.
   - **-d** *local_file*: Name of the local file, including its directory location, you want to copy. Directory path must be absolute.
   - **-s** *remote_file*: Name of the remote file, including its directory location, you want to copy to. Directory path must be absolute.
   - **-p** *consumer*: 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.

     Note:

     If you have a sequence of commands, there is no need log on or provide credentials each time. Note that credentials, which are locally stored, expire after a certain length of time.

- **-u** *user_name*: (Optional.) Fully-qualified user name of the execution account to register the password for.
- **-x** *password*: (Optional.) Password to register for the execution user account.

Note:

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 user_name and password.

Remove a file from a host

You must have appropriate privileges on each of the hosts involved. Directory path names must be absolute. You may wish to first request a resource allocation/slot from EGO before running the commands (if you do not, an allocation is automatically requested on your behalf).

Run the command `rfa remove` to remove a file from a specified host.

1. From the command line, run the `rfa remove` command.

   ```
   rfa remove -t host_name -d local_file | -s remote_file -p consumer [-c
   credential | -u user_name -x password]
   ```

   - **-t** *host_name*: Name of host you want to remove a file from.
   - **-d** *local_file*: Name of the local file, including its directory location, you are removing. Directory path must be absolute.
   - **-s** *remote_file*: Name of the remote file, including its directory location, you are removing. Directory path must be absolute.
   - **-p** *consumer*: 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.

     Note:

     If you have a sequence of commands, there is no need log on or provide credentials each time. Note that credentials, which are locally stored, expire after a certain length of time.
   - **-u** *user_name*: (Optional.) Fully-qualified user name of the execution account to register the password for.
   - **-x** *password*: (Optional.) Password to register for the execution user account.

Note:

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 user_name and password.

Deploying middleware packages

Deploy a middleware package

Middleware packages are copied to the repository server (either to the host on which the `rs` service is running, or to the shared file system if a shared location was configured using `egoconfig mghost`). Middleware packages are then downloaded to hosts and uncompressed.

1. From the command line, run `rsdeploy add` *package_name* -p *package_file* [`-o` *os_type*] [`-n`] [`-f`] [`-u` *user_name*] [`-x` *password*] to add the package to the repository server.

   For example: **rsdeploy add egopackage -p com.platform.ego.updatesite.nt-x86.zip**

   The package gets copied to the repository server host.

   Note:

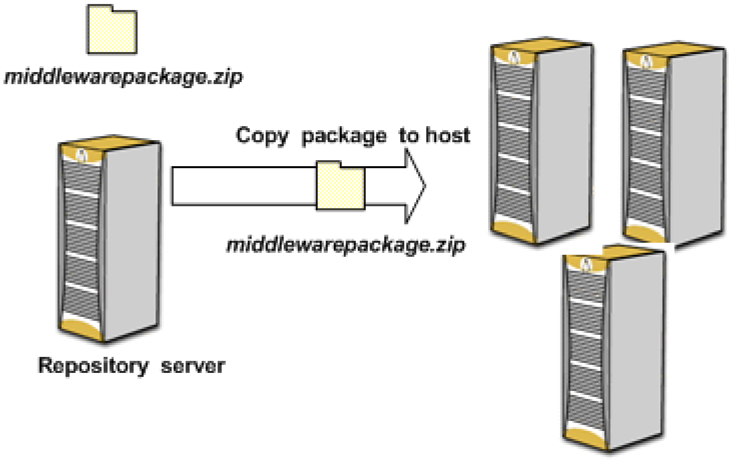
   Package types can be one of the following: `tar.Z`, `tar.gz`, `gz`, `tgz`, taz, `tar`, `jar`, `tar.zip`, or `zip`.

2. Run `rsdeploy install` *package_name* [`-c` *consumer* `-r` *resource_group*] [`-r` *resource_group*] [`-t` *host_name*] [`-f`] [`-u` *user_name*] [`-x` *password*]

   For example: **rsdeploy install egopackage -c /ClusterServices/EGOClusterServices -r ComputeHosts**

   The repository service copies the package from the repository server to the specified compute hosts, and then uncompresses it.

The general installation flow is as follows:



Remove a middleware package

For removal, middleware packages can either be uninstalled from the hosts to which they were deployed, or completely removed from the repository server once they are no longer needed.

1. From the command line, run `rsdeploy uninstall` *package_name* [`-c` *consumer_name* -r *resource_group*] [-r *resource_group*] [`-t` *host_name*] [`-u` *user_name*] [`-x` *password*]

For example: **rsdeploy uninstall egopackage -c /ClusterServices/EGOClusterServices -r ComputeHosts**

The package is uninstalled from hosts in the ComputeHosts resource group.

2. Run `rsdeploy remove` *package_name* [`-o` *os_type*] to remove the package to the repository server.

   For example: **rsdeploy remove egopackage**

   The package is removed from the specified repository server. Once the package is removed from the repository server, you can no longer run `rsdeploy uninstall` to remove the package from a host.

Building your consumer tree

Follow these rules when building a consumer tree:

- You must be a cluster administrator.
- You cannot create a consumer beneath a consumer that has anything registered to it. You must unregister before you can create a sub-consumer.
- You can only create five levels of consumers including the cluster (top) level.
- You cannot change your cluster name.
- You must build your tree from the top down.

Defaults

The consumers ManagementServices and ClusterServices, along with their sub-consumers, are installed by default.

- ManagementServices has one sub-consumer, EGOManagementServices, which runs important system services on management hosts in the cluster. Services include derbydb, plc, purger, RS, ServiceDirector, WEBGUI, and WebServiceGateway. ManagementServices is configured to use the ManagementHosts resource group. Do not modify or delete this consumer.
- ClusterServices is configured to use the InternalResourceGroup resource group.

  It has one sub-consumer, EGOClusterServices, which runs important system services on every host in the cluster. Do not modify or delete ClusterServices or use it to run workload units.

Next step

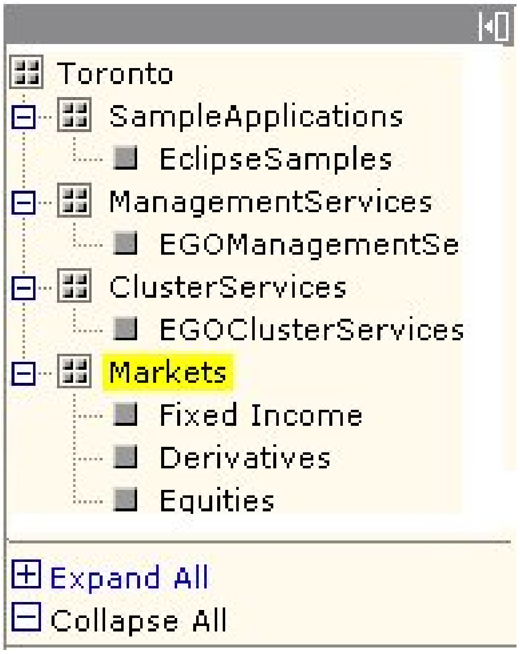
Begin to build your tree by adding consumers.

Example: Build a consumer tree

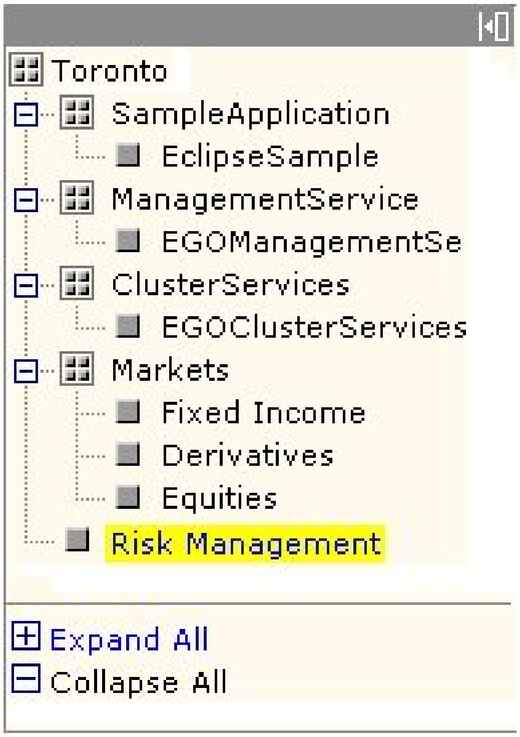
Build your tree to reflect the structure of your business structure.

- Create consumers that reflect your business structure.

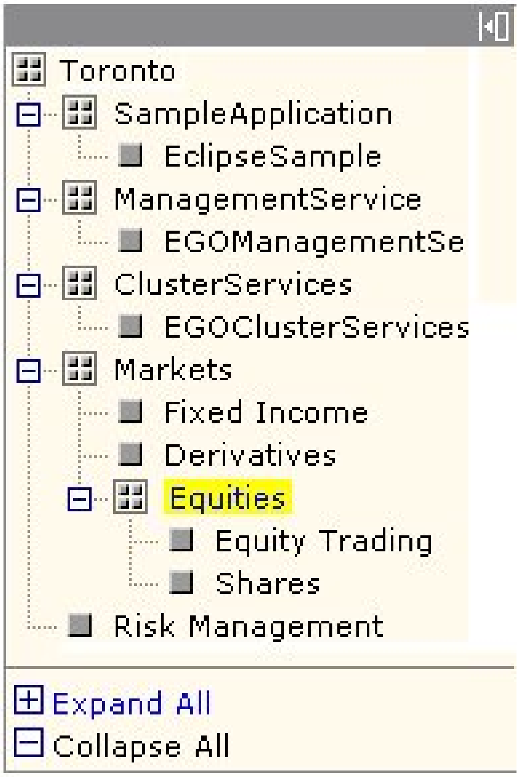
If in the Toronto cluster you have three areas (Fixed Income, Derivatives, and Equities), all reporting to the Markets department who reports to no one, then create your structure as follows:



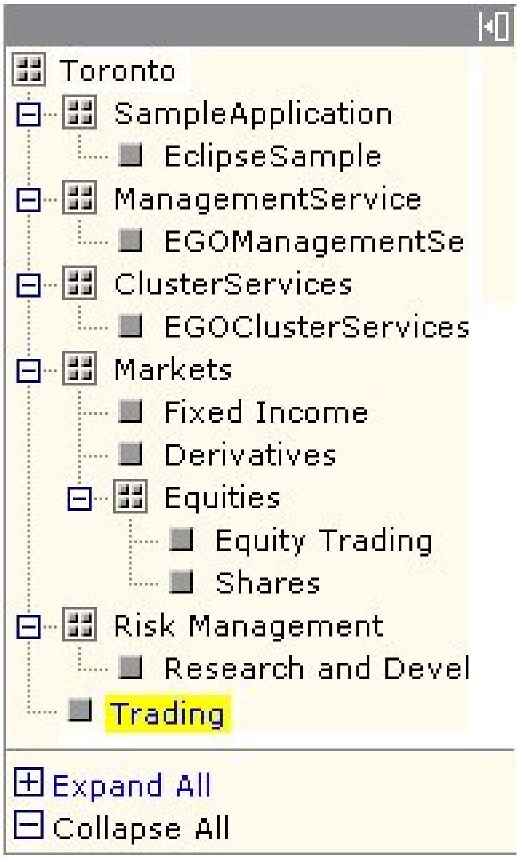
If your company also has the department Risk Management, which reports to no one, add Risk Management to the structure at the same level as Markets:



If Equities is split into Equity Trading and Shares, your structure looks like this:



If you add Trading without any sub-consumers and Research and Development under Risk Management, the consumer tree looks like this:



CHAPTER

3

Managing Services

About system services

A system service runs internal processes for Platform software. System services may have multiple concurrent service instances running on multiple hosts. All system services (except for derbydb) are automatically enabled by default at installation.

By default, management services that are installed with EGO include the following:

- EGO service director (ServiceDirector): A system service that functions as a locating mechanism for other system services.
- Platform Management Console (WEBGUI): A system service that runs the Platform Management Console.
- EGO web service gateway (WebServiceGateway): A system service that provides a standards-based web services interface for web service clients (applications) to contact EGO.
- EGO repository service (RS): A system service that manages package deployment. Allows deployment of a service without reliance on a shared file system.
- PERF data purger (purger): A system service used by the reporting feature.
- PERF loader controller (plc): A system service used by the reporting feature. Manages the data loaders that gather data from the system and writes the data into the database.
- PERF derby database (derbydb): A small-footprint open source database that runs as a system service when first installed. This database is only appropriate for demo clusters. This service is only enabled if an environment variable is set prior to installation (Linux) or during the Windows installation.

By default, system services are registered to the ManagementServices > EGOManagementServices consumer that uses the ManagementHosts resource group. You do not need to register any non-system services to consumers using the ManagementHosts resource group, although it is recommended that you do so if your service is a scheduler or controlling service instead of a regular application (management hosts are not expected to execute workload units for users, but are configured to run important services instead).

Each started system service requires one slot, assuming the system service is configured for one instance (default setting). By default, EGO allots ten slots per management host to run system services, although only six (or seven if derbydb is automatically enabled at installation) are actually used out-of-the-box by the installed management services listed above. For maximum performance in case of failover, you must retain six or seven of the ten slots on each master candidate for use by the system services; the other three can be used by non-system services that you might register that require a management host.

Slots assigned to run system services for consumers the ManagementServices consumer branch do not have to be on the master host (for example, any management host in the cluster can be the web server); the cluster determines where the system service instance runs at startup.

Registered system services do not require a host, only the system service instance that it runs.

Service profile

A service profile is an XML file (configured through the Platform Management Console) that contains the service definition for a service; when you update a system service, you are updating the system service profile containing the system service definition. Every service, whether it is installed by default (a system service) or added by an administrator, has a service profile. The XML file lets you configure properties for the service such as :

- The maximum and minimum number of service instances required by the service

- A service instance description, which describes how to start a service instance for this service
- Any resources required to run the service instances for this service
- Any dependencies this service has (such as another service being started)

Note:

We do not recommend changing the XML file for any of the system services (those installed by default) unless specifically told to do so. You may need to change ports.

Troubleshooting system services

If a system service has the state error or unknown, check the log files to troubleshoot. The logs for the EGO service controller, the load information manager, and the VEM kernel daemon are good places to start.

View a consumer’s system service

You must be a cluster administrator.

You can view the services that you (or the system) associated with a particular consumer. You can also view services for all consumers. Services are associated with consumers because they must exist somewhere in the consumer tree.

1. From System Services, click Monitor Services.

   The Services page displays.
2. In the consumer tree, select the consumer whose services you wish to view.

   Click the cluster name to select from a full list of existing services for all consumers or click a specific consumer name to select from a shorter list of services belonging to only one consumer.

   Note:

   A service can only be registered to a leaf. If your consumer has a sub-consumer, it is not a leaf.

Register a new service

You must be a cluster administrator.

You can register a service if you need to add a new service or if you unregistered a service you need again.Registered services may need to be enabled afterwards if the service is set up to be manually started.

1. From System Services, click Configure Services.

   The Services page displays.
2. In your consumer tree, navigate to the location where you want to register your service.

   You can only register a service to a leaf location. If your consumer has any sub-consumers, then it is not a leaf.
3. From Global Actions, select Register a new service.

   The Service Profile displays showing the minimum required and preferred attributes for a new service along with preconfigured default values.
4. Click Table Preferences at the top of the dialog box and check the boxes to add the corresponding information columns to the table.
5. For main sections in the service profile (for example, "sc:ServiceDefinition"), click Actions and then insert any desired optional service parameters.
6. For each parameter, click within the Value field to add or change the value.
7. Click Register and confirm that you want to modify your service.

Your service is registered. You may see the state change between DEFINED, INIT, ALLOCATING, and STARTED.

If your state remains DEFINED, you need to start your service.

Update a service

You must be a cluster administrator. The service must be in the service state "DEFINED."

Services that are in other service states (such as "STARTED") have service profiles that can be viewed but not edited.

1. From System Services, click Configure Services.

   The Services page displays.
2. In your consumer tree, navigate to the location of the service you wish to update.

   You can only update services from a leaf location. If your consumer has any sub-consumers, then it is not a leaf.
3. Click Actions for the service you wish to edit, and select Open Profile from the drop-down list.

   The Service Profile dialog displays showing the configured parameters for the service.

   Note:

   You can only update services in a DEFINED state. Services in other states are read-only.
4. Click Table Preferences at the top of the dialog box, and check the boxes to add the corresponding information columns to the table.
5. To change any existing parameter values, click within the appropriate Value field and edit the value.
6. To add or remove a parameter, click Actions from the appropriate service section (for example, "sc:ServiceDefinition"), and then insert or remove the desired optional service parameter.
7. Click Save, confirm your changes, and then click Close.

Your service is now updated.

Unregister a service

You must be a cluster administrator.

You can unregister any registered service if you need to troubleshoot it or replace it. By default, EGO services include the ServiceDirector, WEBGUI, and WebServiceGateway services. Do not unregister these services. Unregistering a service means that you can no longer use it. An unregistered service does not appear in the Platform Management Console.

1. From System Services, click Configure Services.

   The Services page displays.
2. In your consumer tree, navigate to the location where the service you want to unregister has been registered.

   Services are only registered and unregistered at a leaf location. If your consumer has any sub-consumers, then it is not a leaf.
3. For the service you want to unregister, click Actions and then select Unregister.
4. Confirm you want to unregister this service.

The service is now unregistered and no longer displays in the Platform Management Console.

Start a service

You must be a cluster administrator. You must register a service before you can start it.

You may have to start a service if you have registered it and have not specified for it to start automatically. The state of your registered service is DEFINED if you need to start it.

1. From System Services, click Monitor Services.

   The Services page displays.
2. In your consumer tree, navigate to the location where the service you want to start is registered or see a list of all registered services by clicking the cluster name.
3. For the service you want to start, click Actions and then select Start.
4. Click OK to confirm that you want to start that service.

Start a service manually

You must be logged on to the command line as a cluster administrator.

You can usually start a service from the Platform Management Console. However, if you have stopped the WEBGUI or you are having trouble accessing the console, you may need to start a service manually.

1. Run **egosh service view** to see the current status of your services.

   If the service you want to start has the status `DEFINED` or `ERROR`, you can proceed to the next step to start it manually.
2. Run **egosh service start *service_name*** where ***service_name*** is the name of the service you want to start.

   Your service starts.

Stop a service

You must be a cluster administrator. A service must be started before you can stop it.

You can stop a service that is running. We do not recommend stopping any system services.

1. From System Services, click Monitor Services.

   The Services page displays.
2. In your consumer tree, navigate to the location where the service you want to start is registered or see a list of all registered services by clicking the cluster name.
3. For the service you want to stop, click Actions and then select Stop.
4. Click OK to confirm that you want to stop that service.

Stop a service manually

You must be logged on as a cluster administrator.

You may need to stop your own (non-EGO) service. This can only be done manually from the command line.

Important:

Do not stop system services. These services are controlled by LIM, which starts and maintains them automatically. Your cluster does not function properly if these services are not running. System services include WEBGUI and SD.

1. From the command line, run **egosh service view** to see the current status of your services.

   If the service you want to stop has the status `INIT`, `ALLOCATING`, or `STARTED`, you can proceed to the next step to stop it manually.
2. Run **egosh service stop -s *service_name*** where ***service_name*** is the name of the service you want to stop.

   Your service stops.

Determine the host address where a system service runs

You need to know the address of where the service director (DNS server) is running (contact IT for assistance, if required).

System services may not all run on the same management host. You can use `nslookup` to find the address of the host where a specific system service is running.

1. Using the CLI, type `nslookup`.
2. Type `server` and then enter the IP address of the service director (DNS server).

   The Default Server and Address returns. For example,

   ```
   > server 179.21.297.3 Default server: 179.21.297.3 Address:
   179.21.297.3#53
   ```
3. Enter the name of the system service for which you want to find the host address. For example,

   ```
   > WEBGUI.egoServer: 179.21.297.3Address: 179.21.297.3#53Name:
   WEBGUI.egoAddress: 180.321.110.34
   ```

Installing and configuring the service director

Overview

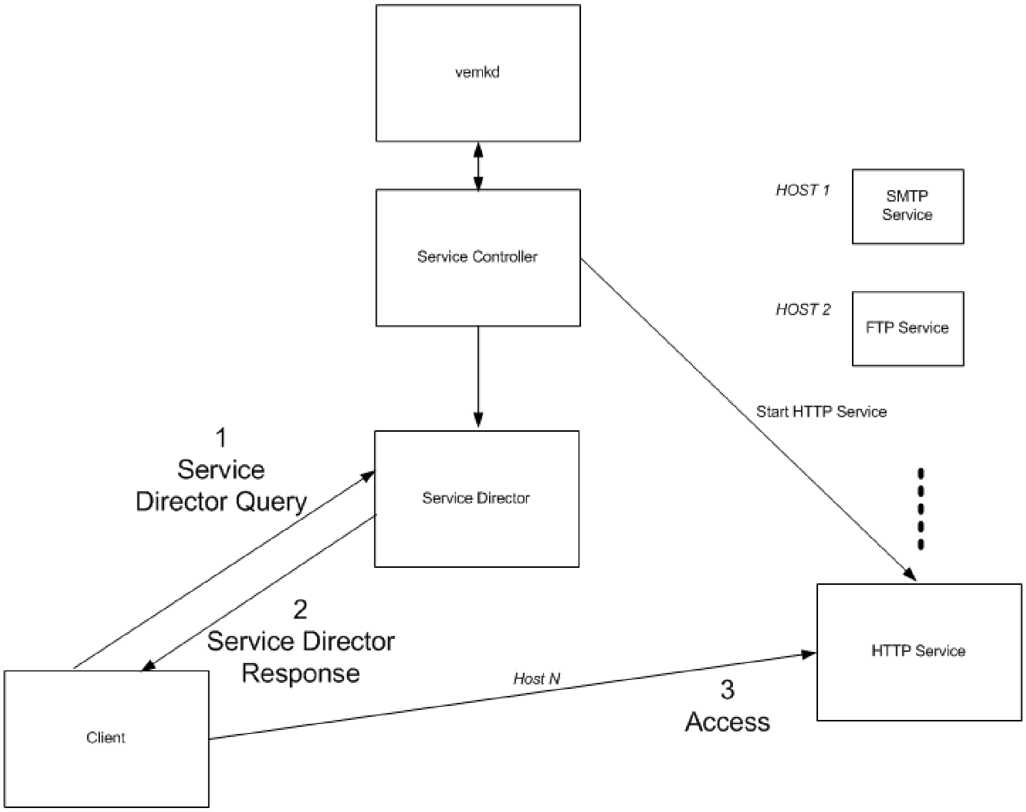
What is the service director?

EGO service director (ServiceDirector) is a service that functions as a locating mechanism for other system services. The service director contains a stand-alone Domain Name Server (DNS), which is the authoritative name server for the EGO DNS sub-domain and responds to DNS queries for system services.

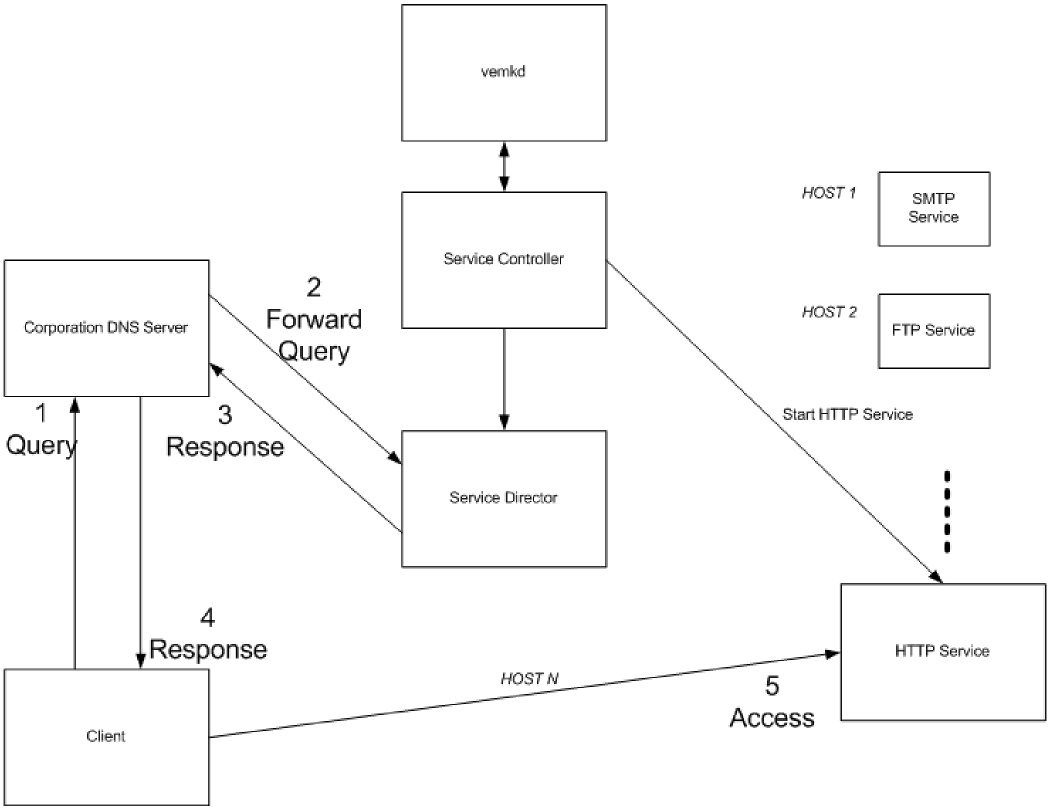
The service director runs on any EGO management host that is the same type as the master, and relies on the service controller to provide location information and state change notifications of service instances.

Why configure the service director?

Out-of-box, a client who is looking to find a host with a EGO service started on it must directly query/point to the service director as nameserver. The nameserver entries get updated each time to point to the service director and to resolve the name and IP address. To avoid making operating system configuration changes each time there is a query, you can configure service director to provide name-to-address mapping to the corporation DNS server (corpdns). Doing this provides location independence for the hosts. You can therefore start a service anywhere.



After configuring the service director, the client can communicate with the corporation DNS server instead, and does not need to be aware of the service director.



About nameserver entries

On Linux, nameserver entries are updated automatically in `/etc/resolv.conf`. On Windows, nameserver entries can be changed interactively, as follows:

1. From the Windows Start menu, click Control Panel, and select Network Connections.
2. Right-click Local Area Connection, and then select Properties.
3. Select Internet Protocol (TCP/IP) from the list of protocols, and then click Properties.
4. On the General page, click Advanced.
5. Click the DNS tab.
6. Add DNS server addresses according to the Windows instructions.

Confirm installation is successful

The candidate host that the service director DNS runs on is automatically selected by EGO during EGO startup. You may wish to confirm ServiceDirector is running, and locate the host it is currently running on.

1. From the command line, run `egosh service list` to confirm ServiceDirector is running.
2. Run `egosh service view` to identify where ServiceDirector is running.

   Look for the address (labelled "Resource") under `Service Instances INFO`. In this example, ServiceDirector runs on "HostA".

```
sh-3.00$ egosh service view
ServiceDirector-----------------------------------------------------------
-------Basic Service INFOService Name       : ServiceDirectorService
Description: EGO: Service Director...Service Instances INFOSTATE
```

```
ACTIVITY_ID      RESOURCE         SEQ_NORUN        4
HostA              1
```

3. Run `dig` to verify that the egonameserver is functioning and the entry for it is in the egonameserver itself.

   For example,

   dig @*HostA* egonameserver.ego A

   where *HostA* refers to the location of the running service director.

Configure the default service director plug-in file

If you want to change the out-of-box service configuration to reflect your organization, you must define certain parameters in the service director default plug-in file.

1. Stop all system services.
2. Open `esddefault.xml`.
   - Windows: *`EGO_TOP`*`\eservice\esd\conf\esddefault.xml`
   - Linux: *`EGO_TOP`*`/eservice/esd/conf/esddefault.xml`

   Note:

   If you ran `egoconfig mghost`, then find `esddefault.xml` in the *EGOshare* directory, where "*EGOshare*" is the shared directory containing important configuration files.

   - Windows: *`EGOshare`*`\eservice\esd\conf\esddefault.xml`
   - Linux: *`EGOshare`*`/eservice/esd/conf/esddefault.xml`
3. Configure the following elements:

Parameter	Description
ESD_EGO_NAMESERVER	The service director DNS server name. (Defining this parameter is optional.) The cluster administrator must add an NS resource record into CORP DNS database, indicating that this server is used as the service director DNS server.
ESD_EGO_DOMAIN	The EGO sub -domain name.
ESD_CORP_DOMAIN	The Corporation domain name.
ESD_EGO_KEY	The TSIG KEY used to update the service director DNS server. (Defining this parameter is optional.)
ESD_CORP_KEY	The TSIG KEY used to update Corporation DNS server. (Defining this parameter is optional.)

4. Restart all system services.

Enable Corp.DNS for external access

By default, Corp.DNS is disabled to limit access to the corporate server from within an internal, corporate domain. If you want to allow for external access, you must enable Corp.DNS and then set up a connection to it. Do this by adding the service director DNS as a sub-domain of Corp.DNS.

1. Open `esddefault.xml`.
   - Windows: *`EGO_TOP`*`\eservice\esd\conf\esddefault.xml`
   - Linux: *`EGO_TOP`*`/eservice/esd/conf/esddefault.xml`
2. Make the following changes:
   - For ESD_EGO_DOMAIN, change **ego** to **service.ego**
   - For ESD_CORP_DOMAIN, change **@EGO_SD_CORPDOMAIN@** to **ego.**

     Also, remove the comment markings for this line.
   - For ESD_CORP_KEY, replace **@EGO_SD_CORPKEYNAME@** with the TSIG key for Corp.DNS (if available); if there is no key available, do not change this line.

   For example:

   ```
   <?xml version="1.0" encoding="UTF-8"?><ESDDefaultPluginConfiguration><!--
   EGO DNS server name --><!-- ESD_EGO_NAMESERVER>@EGO_SD_NAMESERVER@</
   ESD_EGO_NAMESERVER --><ESD_EGO_NAMESERVER>egonameserver</
   ESD_EGO_NAMESERVER><!-- EGO DNS domain name --><!--
   ESD_EGO_DOMAIN>@EGO_SD_EGODOMAIN@</ESD_EGO_DOMAIN -->
   <ESD_EGO_DOMAIN>service.ego</ESD_EGO_DOMAIN><!-- Corporation DNS domain
   name --><ESD_CORP_DOMAIN>ego.</ESD_CORP_DOMAIN>
   <!-- EGO DNS sub-domain TSIG key created by dnssec-keygen --><!--
   ESD_EGO_KEY name="@EGO_SD_EGOKEYNAME@">@EGO_SD_EGOKEY@</ESD_EGO_KEY --
   ><ESD_EGO_KEY name="ego.">rUlWkhrNFCsXkOwZBu/xVA==</ESD_EGO_KEY><!-- CORP
   DNS domain TSIG key created by dnssec-keygen --><!-- ESD_CORP_KEY
   name="@EGO_SD_CORPKEYNAME@">TSIG key</ESD_CORP_KEY --></
   ESDDefaultPluginConfiguration>
   ```
3. Open `named.conf`.
   - Windows: *`EGO_TOP`*`\eservice\esd\conf\named\conf\named.conf`
   - Linux: *`EGO_TOP`*`/eservice/esd/conf/named/conf/named.conf`
4. Make the following changes:
   - For key ego., change **ego.** to **service.ego.**
   - For zone "ego." IN, change **ego.** to **service.ego.**
   - For file "db.ego" (under zone "ego." IN), change **db.ego** to **db.service.ego**

   For example:

   ```
   …key service.ego. {        algorithm HMAC-MD5.SIG-ALG.REG.INT;        secret
   "rUlWkhrNFCsXkOwZBu/xVA==";};…zone "service.ego." IN {        type
   master;        file "db.service.ego";        allow-update { key ego.; };};…
   ```
5. Rename the template file:
   - Windows: *`EGO_TOP`*`\eservice\esd\conf\named\namedb\TMPL.db.EGODOMAIN.CORPDOMAIN\TMPL.db.EGODOMAIN.CORPDOMAIN` to `db.service.ego`
   - Linux: *`EGO_TOP`*`/eservice/esd/conf/named/namedb/TMPL.db.EGODOMAIN.CORPDOMAIN/TMPL.db.EGODOMAIN.CORPDOMAIN` to `db.service.ego`
6. Open the `db.service.ego` template file and make the following changes to it:
   - For @EGO_SD_EGODOMAIN@.@EGO_SD_CORPDOMAIN@., change to **service.ego.**

- For @EGO_SD_NAMESERVER@.@EGO_SD_CORPDOMAIN@., change to **egonameserver.ego.**
- For root.@EGO_SD_EGODOMAIN@.@EGO_SD_CORPDOMAIN@., change to **root.service.ego.**
- For NS @EGO_SD_NAMESERVER@.@EGO_SD_CORPDOMAIN@., change to **egonameserver.ego.**
- For NS @EGO_SD_NAMESERVER@.@EGO_SD_EGODOMAIN@.@EGO_SD_CORPDOMAIN@., change to **egonameserver.service.ego.**

For example:

```
$ORIGIN .$TTL 0  ; 0 seconds service.ego             IN SOA
egonameserver.ego. root.service.ego.
(                            84       ; serial
10800      ; refresh (3 hours)                               900        ;
retry (15 minutes)                               604800     ; expire (1
week)                                 0          ; minimum (0
seconds)                                 )NS
egonameserver.ego.                        NS       egonameserver.service.ego.
```

7. Request that your IT administrator add an NS record in Corp.DNS.

   For example:

   service NS egonameserver

Configure Corp.DNS server

1. If there is a Corp.DNS in your organization, request that your IT administrator add an NS record in it.

   For example:

```
$ORIGIN
$TTL 0  ; 0 secondsego                     IN SOA  ns.ego. root.
(                                69         ; serial                          10800      ;
refresh (3 hours)                                900        ; retry (15
minutes)                                604800     ; expire (1 week)
86400      ; minimum (1 day)                               )                       NS      ns.ego.
$ORIGIN ego.ns                         A       172.17.1.83 (This is the Corp.DNS IP)
service         NS    egonameserver
```

2. If your organization does not have a Corp.DNS, request that your IT administrator create a new `.zone` file on the Corp.DNS server host (for example, `ego.zone`), and then add an NS record in it.
   - Windows: `\var\named\chroot\var\named`
   - Linux: `/var/named/chroot/var/named`
3. Add the following information to `named.conf`:
   - Windows: `EGO_TOP\eservice\esd\conf\named\conf\named.conf`
   - Linux: `EGO_TOP/eservice/esd/conf/named/conf/named.conf`

```
……..key ego. {        algorithm HMAC-MD5.SIG-ALG.REG.INT;        secret
"yOVFmxkFg6eiL4remz5saQ==";};…….zone "ego." IN {        type master;
file "ego.zone";        allow-update { key ego.; };};……
```

Note:

In the command line, run `dnssec-keygen` to generate the "secret" key in the directory `\user\sbin` (Windows) or `/usr/sbin` (Linux). For example: **dnssec-keygen-a HMAC-MD5 -b**

128 -n HOST ego. If a key is not required, add comment markings surrounding the `key ego.` section, and then set `allow-update` to **{ any; };**.

4. Verify the configuration.
   a) From the command line, run `nslookup` to resolve DNS name.

      For example:

      ```
      nslookup > server 172.17.5.82 [This is the Corp.DNS server IP]Default
      server: 172.17.5.82Address: 172.17.5.82#53>
      webgui.service.egoServer:         172.17.5.82  Address:
      172.17.5.82#53Non-authoritative answer:Name:
      webgui.service.egoAddress: 172.17.5.126 [Returns the WEBGUI service
      running host IP]
      ```
   b) From a browser, enter the DNS name as the GUI URL.

      For example: `http:\\webgui.service.ego:8080/platform`.

      Note:

      On EGO clients, you must configure the local host IP to point to Corp.DNS. On Windows hosts, change the IP to the Corp.DNS IP; on Linux hosts, point the IP to Corp.DNS in `resolve.conf`.

Confirm service director DNS server is running

1. Ensure that you have started `ego service ServiceDirector`.
2. Once ServiceDirector is in the RUN state, run `dig` to resolve its name: `egodnsserver.ego.xyz.com`.

   The IP address of the host where ServiceDirector is running should return.

Troubleshoot the service director

Troubleshoot the service director if it fails to update the location information of the DNS server.

1. Check the state of ServiceDirector.

   If it is not in RUN state, the service director does not update the location information of the DNS server.
2. Check for any errors logged in `esc.log`.

   - Windows: `EGO_TOP\eservice\esc\log`
   - Linux: `EGO_TOP/eservice/esc/log`

   For example, you might discover the reason for the failure in an error message like this one: `setServiceInstanceState(): failed to add service director dns server location information(n). host: <h>`

   where `<h>` is hostname, and `(n)` is the error code (which can be looked up in `esd.h`).
3. Check for any errors logged to the Corp DNS server.

   For example, you might find a reason for the failure in an error message like this one: `request has invalid signature:TSIG service.ego: tsig verify failure (BADTIME)`.

   In this case, the Corp DNS server has a different time than that of the EGO master.

About system service instances

System service instances are the result of a running system service. For example, system services that are stopped for any reason do not have system service instances.

A system service instance only runs one activity. You cannot control the activity directly, only the system service instance that contains it. For example, you can migrate an instance (and therefore the activity it contains), but you cannot migrate the activity directly.

You can have multiple system service instances running under a single system service; this means multiple activities can run simultaneously.

By default, system services have a configured number of system service instances to run on each slot (1 instance per slot).

System service instances require a host to run an activity. A system service instance runs using the resource group (for example, the ManagementHosts group or the ComputeHosts group) that is defined in the system services' definition files. Find the definition files here:

- Windows: `EGO_TOP\eservice\esc\conf\services`
- Linux: `EGO_TOP/eservice/esc/conf/services`

Note:

If you ran `egoconfig mghost`, then find the definition file in the *EGOshare* directory, where "*EGOshare*" is the shared directory containing important configuration files.

- Windows: `EGOshare\eservice\esc\conf\services`
- Linux: `EGOshare/eservice/esc/conf/services`

System service instance states

All system service instances have one of the following states:

State	Description
Start	The system service instance is starting.
Run	The system service instance is running.
Finish	The system service instance has completed its task and has finished.
Unknown	The system service instance has not communicated and its current status is unknown. The execution host is unavailable for some reason. The activity is restarted on a new host if HostFailoverInterval is defined and reached.
Zombie	The system service running on an unreachable host has been killed.

A system service instance state is taken from the state of the activity that is running under it.

Control an EGO service instance

EGO service instances are automatically started. You can restart an EGO service instance. You may need to manually start non-EGO services.

- Restart a service instance
- Migrate an EGO service instance

Restart a service instance

You must be a cluster administrator.

You may have to restart a service instance to have a configuration change take effect. For example, you may choose to change your log level for a service. You can restart any service instance through the Platform Management Console.

Restarting a service instance restarts all service instances on the same host (if there are more than one). The system chooses the best host to start the service on again; the system may or may not start the service on the same host as before.

You cannot restart the WEBGUI service instance.

You can restart service instances with the states Start, Run, and Finish.

1. From the System Services page, Monitor Services.

   The Services page displays.
2. Click the service name that runs the service instance you want to restart.

   The Service Properties for that service display.
3. Click Service Instances.

   A list of service instances displays.
4. For the service instance you want to restart, click Actions and then select Restart.

Migrate a service instance

You must be a cluster administrator.

You can migrate service instances to another host if the current host does not meet requirements you have or if you need to take that host down for maintenance.

You cannot select a specific host to migrate to. The system selects the new host automatically. The system blocks the old host from having any new services assigned to it. You can remove a host from the blocked list through the Platform Management Console.

The system also migrates all service instances running on the same host. Migrating service instances takes the service down for a few moments.

You can migrate service instances that have the states Start or Run.

1. From the System Services page, Monitor Services.

   The Services page displays.
2. Click the service name that runs the service instance you want to migrate.

   The Service Properties for that service display.
3. Click Service Instances.

A list of service instances displays.

4. For the service instance you want to migrate, click Actions and then select Migrate.

Your service instance is migrated to a new host. Your old host is placed on the blocked list.

Troubleshoot service error states

If you receive a service error message, or a message indicating that a service cannot transition out of the `Allocating` state, there are steps you can perform to troubleshoot the issue.

- Respond to service message Error
- Respond to service message Allocating

Respond to service message Error

Normally, EGO attempts to start a service multiple times, up to the maximum threshold set in the service profile XML file (containing the service definition). If the service cannot start, you receive a service error message.

1. Try stopping and then restarting the service.
2. Review the appropriate service instance log file to discover the cause of the error.

   System service log files include those for the service director (ServiceDirector), web service gateway (WebServiceGateway), Platform Management Console (WEBGUI), respository service (RS), PERF purger (purger), PERF loader controller (plc), and the PERF derby database service (derbydb). If you have defined your own non-system services, you may have other log files you need to review, depending on the service that is triggering the error.

Respond to service message Allocating

`Allocating` is a transitional service state before the service starts running. If your service remains in this state for some time without transitioning to `Started`, or cycles between `Defining` and `Allocating`, you want to discover the cause of the delay.

1. If you are the cluster administrator, review the allocation policy.
2. Open the service profile from the Platform Management Console. In the sc:AllocationSpecification section, review the consumer for which the service is expected to run.
3. Ensure that a proper resource plan is set for that consumer.

   During a service’s “allocation” period, EGO attempts to find an appropriate resource on which to run the service. If it cannot find the required resource, the service does not start.

CHAPTER

4

Managing Hosts

Important host roles

Hosts in the cluster may be described as the master host, master candidates, management hosts, compute hosts, the web server host, or the DB host.

By default, EGO sets the number of CPUs on hosts by the number of physical CPUs. If you have a dual-core processor, you may wish to change the default behavior and set the number of CPUs by cores on hosts.

Master host

A cluster requires a master host. This is the first host installed. The master host controls the rest of the hosts in the cluster.

Master candidates

There is only one master host at a time. However, if master candidates are specified and the master host ever fails, another host automatically takes over the master host role, allowing work to continue. This process is called failover. When the master host recovers, the role switches back again.

Hosts that can act as the master are called master candidates. This includes the original master host and all hosts that can take over the role in a failover scenario. All master candidates must be management hosts.

Master host failover

During master host failover, the system is unavailable for a few minutes while hosts are waiting to be contacted by the new master.

The master candidate list defines which hosts are master candidates. By default, the list includes just one host, the master host, and there is no failover. If you configure additional candidates to enable failover, the master host is first in the list. If the master host becomes unavailable, the next host becomes the master. If that host is also unavailable, the next host is considered to become the master, and so on down the list. A short list with two or three hosts is sufficient for practical purposes.

For failover to work properly, the master candidates must share a file system and the shared directory must always be available.

Important:

The shared directory should not reside on a master host or any of the master candidates. If the shared directory resides on the master host and the master host fails, the next candidate cannot access the necessary files.

Management host

Management hosts belong to the ManagementHosts resource group. These hosts are not expected to execute workload units for users. Management hosts are expected to run services such as the web server and web services gateway. The master host and all master candidates must be management hosts.

Management hosts share configuration files, so a shared file system is needed among all management hosts.

A management host is configured when you run `egoconfig mghost` on the host. The tag mg is assigned to the management host, to differentiate it from a compute host.

Compute host

Compute hosts are distributed to cluster consumers to execute workload units. By default, compute hosts belong to the ComputeHosts resource group.

The ComputeHosts group excludes hosts with the mg tag, which is assigned to management hosts when you run `egoconfig mghost`. If you create your own resource groups to replace ComputeHosts, make sure they also exclude hosts with the mg tag.

By default, the number of slots on a compute host is equal to the number of CPUs.

Web server host

The web server is the host that runs the Platform Management Console. There is only one web server host; it does not need to be a dedicated host. Any management host can be the web server (decided when the cluster starts up).

DB host

The DB host is the host that runs the database used for Reporting.

For a demo cluster, the Derby database runs as a system service (derbydb) on a management host that you specify during installation. There is no failover, so if this host fails, the Reporting feature does not work properly.

For a production cluster, you must use a supported commercial database, and the host does not need to be part of the cluster.

Host states

Host state	Description
OK	Your host is functioning normally and has no trouble communicating with the system. OK hosts accept new workload. The host is participating in host scavenging. Priority may be set to normal or lowest priority.
Unavailable	One or more of the daemons running on the host have failed or are not communicating with the master host. An unavailable host does not accept new workload.
Closed	A host is closed when an administrator chooses to close or lock it. Administrators close a host to upgrade software or to troubleshoot hardware. Hosts can also be closed when host scavenging has been enabled and the host is no longer idle. A closed host does not accept new workload.

Host properties

Property	Description
Host Name	The name of the host.
Status	The current state of the host: OK, Unavailable, or Closed.
Type (Host Type)	The type of host you have. For example, LINUX86.
CPUs (Number of CPUs)	The number of CPUs you have specified for your host.
CPU Util	The current CPU utilization of your host in %.
Mem (Available Memory)	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.
Swap (Available Swap)	The currently available virtual memory (swap space) in MB. This represents the largest process that can be started on the host.
Pg (Paging Rate)	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.
I/O (Disk I/O Rate)	The I/O throughput to disks attached directly to this host, in KB per second. This rate does not include I/O to disks that are mounted from other hosts.
Slots (Number of Slots)	The number of slots you have specified for this host.
Free Slots (Number of Free Slots)	The number of slots available to run workload units at this time.
15s Load (15-Second Load)	The load this host carries, averaged over the last 15 seconds. The load is the average number of processes using the CPU during a given time interval.
15m Load (15-Minute Load)	The load this host carries, averaged over the last 15 minutes. The load is the average number of processes using the CPU during a given time interval.
1m Load (1-Minute Load)	The load this host carries, averaged over the last minute. The load is the average number of processes using the CPU during a given time interval.
Model (Host Model)	The model of your host. For example, Intel_EM64T.
CPU Factor	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. The CPU factors are defined by the administrator. For multiprocessor hosts, the CPU factor is the speed of a single processor; the system automatically scales the host CPU load to account for additional processors.
Max Mem	The maximum RAM available.
Max Swap	The maximum swap space on your host.

Property	Description
Temp (Available Temp)	The space available in MB on the file system that contains the temporary directory.
Max Temp	Maximum space in `/tmp`.
Disks	Number of local disks on your host.
It (Idle Time)	The number of time in minutes that a host has been idle. On a UNIX host, it is the amount of time since the keyboard has been touched on all logged in sessions. On a Windows host, it is the amount of time a screen saver has been active.
Users (Login Users)	The number of current users logged in to the system.
Resources	If you see mg, the host is a management host.

About load indices

Load indices measure the availability of dynamic resources on hosts. Dynamic resources are properties of a host that change as the load on that host changes, such as available memory and CPU utilization.

Load indices are not configurable; they simply reflect the current state of resources on your hosts given the current load they are managing. For example, as more workload units are assigned to and being run on host A, host A's CPU utilization load index increases. A load index may increase or decrease as the host's resources are put under more load. For example on host A, the total available memory decreases as the load increases.

How often are they measured?

Load indices are measured automatically at fixed time intervals. Each index is individually monitored and has its own update interval (from one of the shortest intervals at 15 seconds for status, to the longest interval at 120 seconds for available temporary space).

Why do I use them?

Viewing load indices for one host provides an excellent snapshot of how that host is performing at a specific moment. For troubleshooting purposes, you may want to track the load indices of one host over time (for example, an hour or a day).

Viewing load indices for all your hosts provides an overall snapshot of how your cluster is performing under its current load at a specific moment.

Load indices

Index	Measures	Units	Direction	Averaged over	Update Interval
status	host status	string			15 seconds
r15s	run queue length	processes	increasing	15 seconds	15 seconds
r1m	run queue length	processes	increasing	1 minute	15 seconds
r15m	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
ls	logins	users	increasing	N/A	30 seconds
it	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
tmp	available space in temporary file system	MB	decreasing	N/A	120 seconds
io	disk I/O	KB per second	increasing	1 minute	15 seconds

Set your master candidates

You must be a cluster administrator.

You can modify the order of the hosts in the candidate list (this is the list that specifies the management host(s) in line to become the new master in situations where the existing master host becomes unavailable). You can also change the master host.

1. Click Cluster > Summary > Master Candidates.
2. To add hosts to your candidate list, select one or more hosts from the available hosts list and click Add.

   Use SHIFT or CTRL while you are clicking to select more than host at a time.
3. To remove any hosts from the candidate list, click the host name and Remove.

   You cannot remove the master host from this list. The master host is at the top of the list by default when you navigate to this page. To remove that host, you must move it down in the list so that it is not the master, and restart your cluster.
4. Change the order of the candidates using Up and Down.

   You can move the master host from the top of the list, however you cannot remove master host from the list.
5. When you have the hosts you want in the order you want, click Apply.

   Changing and saving the order of your candidates restarts your cluster.

The hosts you want as master and master candidates are now set in the order you want them to failover. Your cluster automatically restarts when you click Apply, making the changes take effect.

Control hosts (Windows)

You can start, stop, and restart local, remote, and multiple Windows hosts. Find information on each of these options.

- Start hosts
  a) Start a local Windows host
  b) Start a remote Windows host
  c) Start multiple Windows hosts
- Shut down hosts
  a) Stop a local Windows host
  b) Stop a remote Windows host
  c) Stop multiple Windows hosts
- Enable automatic expiry of unavailable compute hosts
- Remove management hosts from the cluster
- Restart hosts
  a) Restart a local Windows host
  b) Restart a remote Windows host
  c) Restart multiple Windows hosts

Start hosts

Starting hosts brings them into the cluster where they become usable resources.

Start a local Windows host

Log on as egoadmin.

To start a local host, perform the following steps:

1. Start an interactive command console.
2. Start EGO on your local host.

   egosh ego start

Start a remote Windows host

Log on as egoadmin. If the master host is up, log onto any host in the cluster; if the master is down, log onto a management host.

1. Start EGO on your remote host:

   egosh ego start *host_name*

   Replace *host_name* with the name of your remote host.

Start multiple Windows hosts

Log on as egoadmin. If the master host is up, log onto any host in the cluster; if the master is down, log onto a management host.

1. Start EGO on multiple hosts:

   egosh ego start *host_name host_name* ...

Replace *host_name* with the names of your remote hosts. Separate host names with a space.

Shut down hosts

Shutting down a host immediately changes the host state to Unavailable. Running workload is automatically restarted on another host. If you need to perform maintenance, you can choose to close a host instead of shutting it down.

Removing a management host requires following a separate procedure.

Stop a local Windows host

Log on as egoadmin.

1. Stop EGO on your local host:

   egosh ego shutdown

Stop a remote Windows host

Log on as egoadmin. If the master host is up, log onto any host in the cluster; if the master is down, log onto a management host.

1. Stop EGO on your remote host:

   egosh ego shutdown *host_name*

   Replace *host_name* with the name of your remote host.

Stop multiple Windows hosts

Log on as egoadmin. If the master host is up, log onto any host in the cluster; if the master is down, log onto a management host.

1. Stop EGO on more than one host:

   egosh ego shutdown *host_name host_name* ...

   Replace *host_name* with the names of your remote hosts. Separate host names with a space.

Enable automatic expiry of unavailable compute hosts

By default, all hosts that join the cluster remain in the cluster, even if they become unusable.

Optionally, after a period of time in the Unavailable state, a compute host can expire from the cluster—it no longer appears in the Console, or in `egosh resource list` or `resource view` output, and it is not affected by `egosh start all` or `restart all` commands.

Host expiry is not irreversible. If you restart an expired host (for example, run `egosh ego start` *host_name* or restart the host while automatic system startup is configured), it can rejoin the cluster in the same way that a newly installed host joins the cluster.

If you want to remove a management host from the cluster, a different procedure is required.

To configure the host expiry feature, take the following steps.

1. Logon as egoadmin and edit `ego.conf`.
2. Specify a time-out period for the parameter EGO_DYNAMIC_HOST_TIMEOUT.

   The default time period is in hours. Use an M after the time value to represent minutes (for time periods of 10 minutes or more).

For example:

- `EGO_DYNAMIC_HOST_TIMEOUT=48` means the unavailable host is removed after 48 hours.
- `EGO_DYNAMIC_HOST_TIMEOUT=75M` means the host is removed after an hour and fifteen minutes.
- `EGO_DYNAMIC_HOST_TIMEOUT=2M` means the host is removed after 10 minutes; a value of 2 minutes is below the allowable 10 minute minimum.

3. Restart the master host for the change to take effect.

Remove management hosts from the cluster

You must have previously enabled automatic expiry of unavailable compute hosts.

Hosts that have been added to a cluster using the `egoconfig mghost cmd` and are designated as management hosts need to be physically deleted from `ego.cluster.`*`cluster_name`* if you want to remove them from the cluster. This requirement applies to current management hosts.

1. Shut down the host.
2. If you have configured automatic startup during your cluster setup, then run `egoremoverc.sh`.

   Doing this prevents automatic startup when the host reboots, which keeps the host from being re-added dynamically to the cluster.
3. If the host is a master candidate, run `egoconfig masterlist` to remove the host from the failover order.
4. Run `egoconfig unsetmghost` to remove the host from the management host group.

   Running this command removes the host entry from ego.cluster.*cluster_name*.
5. Run the installer (`MSI` file) on the host you wish to remove from the cluster, and uninstall the EGO package.
6. Restart the master host to change it from a management host to a compute host.

   Because the host is shut down, and daemons are no longer running, the host switches to an unavailable state. Now that you've got a compute host in an unavailable state, complete the steps for enable automatic expiry of unavailable compute hosts to remove the host from the cluster.

Restart hosts

You may want to restart a host if it has become unavailable to the cluster. An unavailable host may have problems with memory or unnecessary applications that can be fixed by restarting it.

Restart a local Windows host

Log on as egoadmin.

1. Restart EGO on your local host:

   egosh ego restart

Restart a remote Windows host

Log on as egoadmin. If the master host is up, log onto any host in the cluster; if the master is down, log onto a management host.

1. Restart EGO on your remote host:

   egosh ego restart *host_name*

   Replace *host_name* with the name of your remote host.

Restart multiple Windows hosts

Log on as egoadmin. If the master host is up, log onto any host in the cluster; if the master is down, log onto a management host.

1. Restart EGO on multiple hosts:

   egosh ego restart *host_name* *host_name* ...

   Replace *host_name* with the names of your remote hosts. Separate host names with a space.

Control hosts (Linux)

You can start, shut down, and restart local, remote, or multiple Linux hosts. Find information on each of these options.

- Start hosts
  a) Start a local Linux host
  b) Start a remote Linux host
  c) Start multiple Linux hosts
- Shut down hosts
  a) Stop a local Linux host
  b) Stop a remote Linux host
  c) Stop multiple Linux hosts
- Enable automatic expiry of unavailable compute hosts
- Remove management hosts from the cluster
- Restart hosts
  a) Restart a local Linux host
  b) Restart a remote Linux host
  c) Restart multiple Linux hosts

Start hosts

Log on with root permissions and rsh on all hosts in the cluster. To start a 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.

Important:

By default, only root can start, stop, or restart the cluster. Optionally, you can grant root privileges to egoadmin, the cluster administrator account.

Starting hosts brings them into the cluster where they become usable resources.

Start a local Linux host

Log on with root permissions.

To start a local host, take the following steps.

1. Start EGO on your local host:

   egosh ego start

Start a remote Linux host

Log on with root permissions and rsh on the local host and the remote host. If the master is up, log onto any host in the cluster; if the master is down, log onto a management host.

1. Start EGO on a remote host:

   ego start *host_name*

   Replace *host_name* with the name of your remote host.

Start multiple Linux hosts

Log on with root permissions and rsh on the local host and each remote host. If the master is up, log onto any host in the cluster; if the master is down, log onto a management host.

1. Start EGO on multiple hosts:

   `ego start` ***host_name1 host_name2 ...***

   Replace *host_name* with the names of your remote hosts. Separate host names with a space.

Shut down hosts

Shutting down a host immediately changes the host state to Unavailable. Running workload is automatically restarted on another host. If you need to perform maintenance, you can choose to close a host instead of shutting it down.

Stop a local Linux host

Log on with root permissions.

1. Stop EGO on your local host:

   egosh ego shutdown

Stop a remote Linux host

Log on with root permissions on the local host. If the master host is up, log onto any host in the cluster; if the master is down, log onto a management host.

1. Stop EGO on a remote host:

   `egosh ego shutdown` ***host_name***

   Replace *host_name* with the name of your remote host.

Stop multiple Linux hosts

Log on with root permissions on the local host. If the master host is up, log onto any host in the cluster; if the master is down, log onto a management host.

1. Stop EGO on multiple hosts:

   `egosh ego shutdown` ***host_name1 host_name2 ...***

   Replace *host_name* with the names of your remote hosts. Separate host names with a space.

Enable automatic expiry of unavailable compute hosts

By default, all hosts that join the cluster remain in the cluster, even if they become unusable.

Optionally, after a period of time in the Unavailable state, a compute host can expire from the cluster—it no longer appears in the Console, or in `egosh resource list` or `resource view` output, and it is not affected by `egosh start all` or `restart all` commands.

Host expiry is not irreversible. If you restart an expired host (for example, run `egosh ego start` *host_name* or restart the expired host while automatic system startup is configured), it can rejoin the cluster in the same way that a newly installed host joins the cluster.

If you want to remove a management host from the cluster, a different procedure is required.

To configure the host expiry feature, take the following steps.

1. Logon as egoadmin and edit `ego.conf`.
2. Add the parameter EGO_DYNAMIC_HOST_TIMEOUT and specify an expiry period.

   The default time period is in hours. Use an M after the time value to represent minutes (for time periods of 10 minutes or more).

   For example:

   - `EGO_DYNAMIC_HOST_TIMEOUT=48` means the unavailable host is removed after 48 hours.
   - `EGO_DYNAMIC_HOST_TIMEOUT=75M` means the host is removed after an hour and fifteen minutes.
   - `EGO_DYNAMIC_HOST_TIMEOUT=2M` means the host is removed after 10 minutes; a value of 2 minutes is below the allowable 10 minute minimum.
3. Restart the master host for the change to take effect.

Remove management hosts from the cluster

You must have previously enabled automatic expiry of unavailable compute hosts.

Hosts that have been added to a cluster using the command `egoconfig mghost`, and are designated as management hosts, need to be physically deleted from `ego.cluster.`*`cluster_name`* if you want to remove them from the cluster. This requirement applies to current management hosts.

1. Shut down the host.
2. If you have configured automatic startup during your cluster setup, then run the command `egoremoverc.sh` as root.

   Doing this prevents automatic startup when the host reboots, and keeps the host from being re-added dynamically to the cluster.
3. If the host is a master candidate, run the command `egoconfig masterlist` to remove the host from the failover order.
4. Run `egoconfig unsetmghost` to remove the host from the management host group.

   Running this command removes the host entry from `ego.cluster.`*`cluster_name`*.
5. Optional. If you want to completely remove the host from the cluster, you can shutdown all the applicable services and daemons at this point, and then remove the directory where the Linux installation is located.
6. Restart the master host to change it from a management host to a compute host.

   Because the host is shut down, and daemons are no longer running, the host switches to an unavailable state. Now that you’ve got a compute host in an unavailable state, complete the steps for Enable automatic expiry of unavailable compute hosts.

Restart hosts

You may want to restart the EGO daemons on a host if it is in an Unavailable state. An unavailable host may have problems with memory or unnecessary applications that can be fixed by restarting it.

Restart a local Linux host

Log on with root permissions.

1. Restart EGO on your local host:

   egosh ego restart

Restart a remote Linux host

Log on with root permissions. If the master host is up, log onto any host in the cluster. If the master is down, log onto a management host.

1. Restart EGO on a remote host:

   egosh ego restart *host_name*

   Replace *host_name* with the name of your remote host.

Restart multiple Linux hosts

Log on with root permissions. If the master host is up, log onto any host in the cluster; if the master is down, log onto a management host.

1. Restart EGO on multiple hosts:

   egosh ego restart *host_name1 host_name2* ...

   Replace *host_name* with the names of your remote hosts. Separate host names with a space.

Open a host

The host must belong to the cluster and be closed before you can open it. Only administrators can open hosts.

You may need to run `egosetsudoers.sh` on the host running the Platform Management Console processes to be able to open and close hosts from the Platform Management Console.

You can close a host without removing it from the cluster if you need to run diagnostic tests or troubleshoot. It can then be opened again from the Platform Management Console. Closed hosts do not accept new workload. Take a host or hosts down so that you can perform maintenance on it and return it to the cluster. Closing a host means that EGO does not allocate this resource to run workload until it is opened again.

1. Click Resources > Monitor/Control Hosts > Hosts (List View).

   The Hosts (List View) page displays.
2. Locate the host you want to open.

   Use Filter Settings to narrow down your search quickly, to sort, or to search for a specific host by name.
3. For the host you want to open, select Actions > Open.

   If you used filter settings to display hosts in a specific state, the host may not display in the list. Change your filter settings to display hosts in the state you want.

Close a host

The host must belong to the cluster and be open before you can close it. Only administrators can open and close hosts.

You may need to run `egosetsudoers.sh` on the host running the Platform Management Console processes to be able to open and close hosts from the Platform Management Console.

You can close a host without removing it from the cluster if you need to run diagnostic tests or troubleshoot. It can then be opened again from the Platform Management Console. Closed hosts do not accept new workload, but they do continue to run existing work. Close a host so that you can perform maintenance on it and return it to the cluster.

Once a host is closed, EGO does not re-allocate it. Closing a host does not change its allocation status. If the host is currently allocated to a consumer, the host remains allocated until the consumer returns it voluntarily. If the host is not currently allocated to a consumer, the host remains in its unallocated state.

1. Click Resources > Monitor/Control Hosts > Hosts (List View).

   The Hosts (List View) page displays.
2. Locate the host you want to close.

   Use Filter Settings to narrow down your search quickly, to sort, or to search for a specific host by name.
3. For the host you want to close, select Actions > Close.
4. Indicate if you want to reclaim work on this host before closing it.

   If you do not reclaim the work, the host continues to run the work until it is complete before closing.

Your host closes and the status of the host changes to CLOSED, with one of the following reasons:

- Host is unlicensed.
- Host is locked by administrator.
- Host is locked by time window.
- Host is locked by master.
- Host is locked due to no CPU.
- Host is busy due to load.
- Host is closed by administrator.
- Cluster administrator closes and reclaims host.

If you use filter settings to display hosts in a specific state, the host may not display in the list. Change your filter settings to display hosts in the state you want.

You can open the host again at any time. A closed host does not accept any additional requests until it is opened again. The host still belongs to the cluster but now its state is Closed.

Monitor host health from the dashboard

You can only access the dashboard if you are logged in as a cluster administrator or a consumer administrator.

You can get complete information about hosts from the dashboard.

1. Use your dashboard to monitor the hosts in your cluster.

   The pie chart gives you visual cues about the ratio of the states of all the hosts in your cluster.

   The CPU Utilization chart gives you a visual summary of the hosts in your cluster by how may hosts share CPU utilization ranges.
2. From the dashboard, in the Cluster Health area, click one of the states under Hosts by State (for example, the host state OK).

   The Hosts (Icon View) page displays the information about the hosts in that particular state.

   To view all hosts (regardless of state), click the heading Hosts by State.

   Note:

   If you have previously filtered hosts on the Hosts (Icon View) or Hosts (List View) pages, clicking **Hosts by State** only returns those hosts matching the filtered state.
3. Locate the host you want information about.

   Use Filter Settings to search for a specific host by name or to narrow down your search quickly by filtering the hosts by type, state, process priority or CPU utilization.
4. Rest your mouse over the icon to display abbreviated information about the host.
5. For more information about a particular host, click the host icon or toggle to Hosts (List View).

   You can sort hosts by clicking on a column header or using Preferences.

   You can also retrieve and view log files located on a particular host by clicking the corresponding Actions > Retrieve Logs.

Detect when failover occurs

You may need to know when a failover occurs so you can troubleshoot your system or your hosts.

1. Edit `ego.conf`.

   You need to specify two parameters under the EGO event configuration section: EGO_EVENT_PLUGIN and EGO_EVENT_MASK. These parameters are already included in `ego.conf` with default values, but may be commented out.
2. Set up an SNMP v1 (Simple Network Management Protocol version 1) trap for EGO events.
   a) Specify the name and configuration file location with your SNMP information. The plug in name should not include a suffix (.dll or .so): EGO_EVENT_PLUGIN=plugin_name[plugin_conf]…

      Example:

      `EGO_EVENT_PLUGIN=eventplugin_snmp[SINK=`*host*,MIBDIRS=*EGO_TOP*/mibs]

   (where *host* represents the name of the host where the SNMP trap daemon is running).

   SNMP traps enable an agent to notify the management station of significant events by way of an unsolicited SNMP message.

   Note the following:

   - The MIBDIRS directory may also equal *`$EGO_CONFDIR`*`/kernel/conf/mibs`.
   - In a Windows environment, use quotation marks around the event plug-in definition.

     For example, `EGO_EVENT_PLUGIN="eventplugin_snmp [SINK=`*host*,MIBDIRS=*EGO_TOP*\mibs]"
3. Set EGO_EVENT_MASK=LOG_INFO.

   SYS_VEMKD_UP (an INFO level log entry) occurs when you reconfigure your cluster or when your master host has successfully failed over to a new host.
4. Save `ego.conf` and restart your cluster.
5. Using your SNMP Manager, specify what action you want to take when the trap returns a SYS_VEMKD_UP event.

   You can select what actions the SNMP Manager takes when it receives a specific trap. See your SNMP Manager help for more information.

You are notified of a SYS_VEMKD_UP event that occurs only during successful failover and a cluster reconfiguration.

Note:

To verify that the master host has failed over, run `egosh resource list -m`. Your master candidates display. Check the Current Master entry.

Multi-homed hosts

Hosts that have more than one network interface usually have one Internet Protocol (IP) address for each interface. Such hosts are called multi-homed hosts.

EGO identifies hosts by their official host name, so it needs to match each of the network addresses of multi-homed hosts with a single host name. To do this, the host name information must be configured so that all of the Internet addresses for a host resolve to the same name.

Multiple network interfaces

Some system manufacturers recommend that each network interface, and therefore, each Internet address, be assigned a different host name. Each interface can then be directly accessed by name. This setup is often used to make sure NFS requests go to the nearest network interface on the file server, rather than going through a router to some other interface. Configuring this way can confuse EGO, because there is no way to determine that the two different names (or addresses) mean the same host.

All host naming systems can be configured so that host address lookups always return the same name, while still allowing access to network interfaces by different names. Each host has an official name and a number of aliases, which are other names for the same host. By configuring all interfaces with the same official name but different aliases, you can refer to each interface by a different alias name while still providing a single official name for the host.

IP connectivity

Some or all hosts have multiple network interfaces that connect to physically segmented networks. You may not want EGO to use the first IP address according to DNS to initiate a connection.

IP preference

A host has multiple network interfaces that connect to physically connected networks, but for routing or performance reasons, you might want to assign network interface preferences to different activities.

For example, communication between a Platform Symphony client and management hosts could use one network interface, and communication between a Platform Symphony compute and management hosts could use another network interface. While it might be physically possible for a socket client to use the first IP address of a socket server according to DNS to initiate a connection, this might not be desirable.

Host name lookup

A common DNS server may return a different IP address in host name lookups depending on which subnet that host is on (different BIND/DNS views). For example, host named hostA might resolve to 192.168.0.1 on one subnet and 10.0.0.1 on another subnet on the same network.

Filtering a preferred IP address from multiple IP addresses

Use EGO_PREFERRED_IP_MASK in `ego.conf` to specify the preferred IP address for multiple network interfaces.

If more than one IP address matches the IP mask, the first matching IP address is used as the preferred IP address. If no addresses match the mask, the order of the address list is not changed.

Under some circumstances (when you have multiple aliases), you also need to specify the unique official name and list the aliases.

IP mask format

Specify the IP mask as conventional CIDR blocks, consisting of a four-part dotted-decimal address, followed by a slash, then a number from 0 to 32:

`nnn.nnn.nnn.nnn/nn`

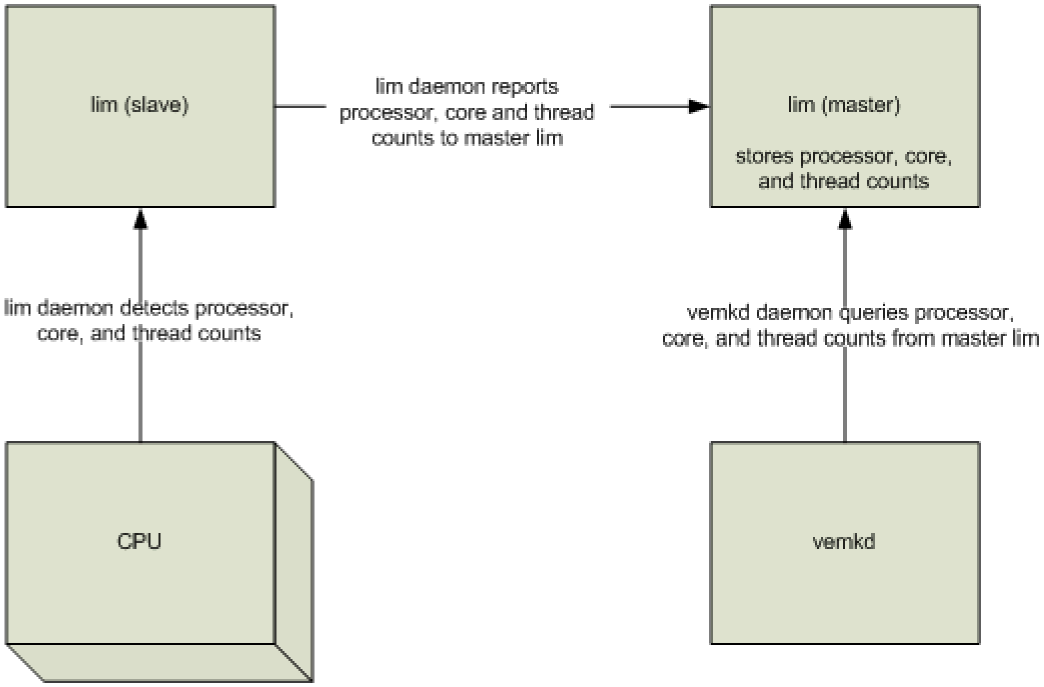
Understanding how the lim daemon detects cores, threads, and processors

Traditionally, the value of ncpus has been equal to the number of physical CPUs. However, most CPUs consist of multiple cores and threads, so the traditional 1:1 mapping is no longer useful. A more useful approach is to set ncpus to equal one of the following:

- The number of processors (this is the ncpus default setting)
- Cores: the number of cores (per processor) * the number of processors
- Threads: the number of threads (per core) * the number of cores (per processor) * the number of processors

An EGO cluster administrator globally defines how ncpus is computed using the EGO_DEFINE_NCPUS parameter (instead of EGO_ENABLE_DUALCORE) in `ego.conf` (shared directory).

The lim daemons detect and store the number of processors, cores, and threads for all supported architectures. The following diagram illustrates the flow of information between daemons, CPUs, and other components.


lim (slave)
lim daemon reports processor, core and thread counts to master lim
lim (master)
stores processor, core, and thread counts
lim daemon detects processor, core, and thread counts
vemkd daemon queries processor, core, and thread counts from master lim
CPU
vemkd

Although the ncpus computation is applied globally, it can be overridden on a per-host basis.

To correctly detect processors, cores, and threads, lim daemons/services assume that all physical processors on a single machine are of the same type.

In cases where CPU architectures and operating system combinations may not support accurate processor, core, thread detection, lim uses the defaults of 1 processor, 1 core per physical processor, and 1 thread per core. If lim detects that is it is running in a virtual environment (for example, VMware®), each detected processor is similarly reported (as a single-core, single-threaded, physical processor).

Lim detection uses processor- or OS-specific techniques (for example, Intel’s CPUID instruction, or Solaris’ kstat()/core_id). Note that if the operating system doesn't recognize a

CPU or core (for example, if an older OS does not recognize a quad-core processor and instead detects it as dual-core), then the lim won't recognize it either; the lim only detects hardware that is recognized by the operating system.

Note:

RQL normalization never considers threads. Consider a hyper-thread enabled Pentium: Threads are not full-fledged CPUs, so considering them as CPUs would artificially lower the system load.

On a machine running AIX, detection of ncpus is different. Under AIX, the number of detected physical processors is always 1, whereas the number of detected cores is always the number of cores across all physical processors. Thread detection is the same as other operating systems (the number of threads per core).

Change default behavior of ncpus: Set the number of CPUs by cores

By default, EGO sets the number of CPUs on hosts by the number of physical CPUs. However, if you have a dual-core processor, you can change this default.

1. Open `ego.conf` for editing.
2. Set EGO_DEFINE_NCPUS=**cores**.
3. Save and close the file.
4. Restart EGO.

Define ncpus—processors, cores, or threads

An EGO cluster administrator must define how ncpus is computed. Usually, the number of available EGO slots is equal to the value of ncpus; however, slots can be redefined at the resource group level. The ncpus definition is globally applied across the cluster.

1. Open `ego.conf`.
    - Linux: `$EGO_CONFDIR/kernel/conf/ego.conf`
    - Windows: `%EGO_CONFDIR%\kernel\conf\ego.conf`
2. Define the parameter EGO_DEFINE_NCPUS=[procs | cores | threads].

    Set it to one of the following:

    - procs (where ncpus=procs)
    - cores (where ncpus=procs *cores)
    - threads (where ncpus=procs *cores * threads)

    By default, ncpus is set to procs (number of processors).

    Note:

    In clusters with older lim daemons/services that do not recognize cores and threads, this parameter is ignored. In clusters where only the master lim recognizes cores and threads, the master lim assigns defaults values (for example, in Platform LSF 6.2: 1 core, -1 thread).
3. Save and close `ego.conf`.

Note:

As a best practice, set EGO_DEFINE_NCPUS instead of EGO_ENABLE_DUALCORE. The functionality of EGO_ENABLE_DUALCORE=y is preserved by setting EGO_DEFINE_NCPUS=cores.

Interacting with related LSF parameters

If your cluster is running the Platform LSF application manager on EGO, there is a duplicate parameter in `lsf.conf` called LSF_ENABLE_DUALCORE. In the case where both EGO and LSF parameters provide a definition for ncpus, the EGO parameter takes precedence.

Override the global configuration of ncpus computation

An EGO cluster administrator globally defines how ncpus is computed. The ncpus global definition can, however, be overridden on specified dynamic and static hosts in the cluster.

- Defining computation of ncpus on dynamic hosts
- Defining computation of ncpus on static hosts

Defining computation of ncpus on dynamic hosts

1. Open `ego.conf`.
    - Linux: `$EGO_CONFDIR/kernel/conf/ego.conf`
    - Windows: `%EGO_CONFDIR%\kernel\conf\ego.conf`
2. Define the parameter EGO_LOCAL_RESOURCES="[**resource *resource_name***]".

    Set *resource_name* to one of the following:

    - define_ncpus_procs
    - define_ncpus_cores
    - define_ncpus_threads

    Note:

    Resource definitions are mutually exclusive. Choose only one resource definition per host.

    For example:

    - Windows: EGO_LOCAL_RESOURCES="[type NTX86] [resource define_ncpus_procs]"
    - Linux: EGO_LOCAL_RESOURCES="[resource define_ncpus_cores]"
3. Save and close `ego.conf`.

Note:

In multi-cluster environments, if ncpus is defined on a per-host basis (thereby overriding the global setting) the definition is applied to all clusters that the host is a part of. In contrast, globally defined ncpus settings only take effect within the cluster for which EGO_DEFINE_NCPUS is defined.

Defining computation of ncpus on static hosts

1. Open `ego.cluster.`*`cluster_name`*.
    - Linux: `$EGO_CONFDIR/kernel/conf/ego.cluster.`*`cluster_name`*
    - Windows: `%EGO_CONFDIR%\kernel\conf\ego.cluster.`*`cluster_name`*
2. Find the host for which you want to define ncpus computation. In the RESOURCES column, add one of the following definitions:
    - define_ncpus_procs
    - define_ncpus_cores
    - define_ncpus_threads

    Note:

    Resource definitions are mutually exclusive. Choose only one resource definition per host.

For example:

```
Begin HostHOSTNAME  model    type        r1m  mem  swp  RESOURCES
#Keywords#lemon    PC200    LINUX86     3.5  1    2   (linux)
#plum     !        NTX86       3.5  1    2   (nt)Host_name !
NTX86        -   -    -   (define_ncpus.procs)End     Host
```

3. Save and close `ego.cluster.`*`cluster_name`*.
4. Restart the master host.

Note:

In multi-cluster environments, if ncpus is defined on a per-host basis (thereby overriding the global setting) the definition is applied to all clusters that the host is a part of. In contrast, globally defined ncpus settings only take effect within the cluster for which EGO_DEFINE_NCPUS is defined.

Understanding RQL normalization between Platform LSF and EGO

Mapping slots in a way that considers both threads and cores affects RQL normalization (a part of the load-level computation) across hosts in a cluster. Because the computed load-level may be artificially lowered, RQL normalization never considers threads. By default, however, RQL normalization does not consider the number of cores, either. This default behavior can be changed.

Interacting with related LSF parameters

If your cluster is running the Platform LSF application manager on EGO, there are parameters that interact with and/or duplicate the functionality of EGO_DEFINE_NCPUS:

- LSF_ENABLE_DUALCORE: If set to "cores," this parameter is equivalent to EGO_DEFINE_NCPUS. They both define how ncpus is computed.

If EGO_DEFINE_NCPUS is set to...	Then RQL normalization considers...
procs	processors
cores	processors and cores
threads	processors and cores, but not threads

Note:

In cases where both EGO and LSF parameters provide definitions, the EGO parameter takes precedence.

- LSF_RQL_CONSIDER_DUALCORE: This parameter instructs the lim to consider the number of cores when computing the RQL normalization. It is set in conjunction with LSF_ENABLE_DUALCORE. Note the following interactions:
    - If both LSF_ENABLE_DUALCORE and LSF_RQL_CONSIDER_DUALCORE are set, then RQL normalization takes into account both cores on a machine with dual-core processors.
    - If LSF_RQL_CONSIDER_DUALCORE is set, but LSF_ENABLE_DUALCORE is not, then LSF_RQL_CONSIDER_DUALCORE is ignored.
    - If EGO_DEFINE_NCPUS is set, then LSF_RQL_CONSIDER_DUALCORE option is always ignored. (RQL normalization always considers foremost how ncpus is defined by the EGO cluster administrator.)

View host models and types

View detected host models and types

1. Run `egosh resource view [resource_name …]` to display information about host models that exist in the cluster.

View UNKNOWN and DEFAULT host models and types

Viewing UNKNOWN host type or model

1. Run `egosh resource view [resource_name …]`.

   A model or type UNKNOWN indicates the host is down or the lim on the host is down. You need to take immediate action.

Viewing DEFAULT host type or model

1. Run `egosh resource view [resource_name …]`. If model or type are displayed as DEFAULT and automatic host model and type detection is enabled, you can leave it as is or change it.

   If model is DEFAULT, EGO works correctly but the host has a CPU factor of 1, which may not make efficient use of the host model.

   If type is DEFAULT, there may be binary incompatibility. For example, there are two hosts, one is Solaris, the other is HP. If both hosts are set to type DEFAULT, it means jobs running on the Solaris host can be migrated to the HP host and vice versa.

Fix UNKNOWN or DEFAULT Matched Models and Matched Types

Fixing UNKNOWN Matched Type or Matched Model

A model or type UNKNOWN indicates the host or lim on the host is down. You need to take immediate action.

1. Start the host.
2. With root (Unix) or administrator (Windows) permission, run `egosh ego start` *host_name* to start up the load information manager (lim) on the host.

   You can specify more than one host name to start up the lim on multiple hosts. If you do not specify a host name, the lim is started up on the host from which the command is submitted.

   You must be a cluster administrator to run this command.

   On UNIX, to start up the lim remotely, you must be root or listed in `ego.sudoers` and be able to run the `rsh` command across all hosts without entering a password.
3. Wait a few seconds, then run `egosh resource view [resource_name …]`.

   You should now be able to see either a matched model or type for the host or the result DEFAULT. If you see DEFAULT, it means that automatic detection of host type or model has failed, and the host type configured in `ego.shared` cannot be found. EGO still works on the host, but there are disadvantages:

   - A DEFAULT Matched Type may cause binary incompatibility because a job from a DEFAULT host type can be migrated to another.
   - A DEFAULT Matched Model may be inefficient because of incorrect CPU factors.

Fixing DEFAULT Matched Type or Matched Model

If automatic detection of host type or model fails, and the host type configured in `ego.shared` cannot be found, then Matched Type gets set to DEFAULT. A Matched Type reported as DEFAULT may contribute to binary incompatibilities; a Matched Model reported as DEFAULT may be inefficient due to an incorrect CPU factor. You can run `lim -t` to detect the real type or model for a host, and then make changes to `ego.shared`.

1. Run `lim -t` on the host whose type is DEFAULT.
2. Edit `ego.shared`.
   a) In the HostType section, enter a new host type. Use the host type name detected with `lim -t`.
   b) In the HostModel section, add the new model with architecture and CPU factor. Add the host model to the end of the host model list. The limit for host model entries is 127. Lines commented out with # are not counted as part of the 127 line limit.

      Use the architecture detected with `lim -t`.
3. Save changes to `ego.shared`.
4. Run `egosh ego restart` on master host.
5. Wait a few seconds, then run `egosh resource view [resource_name …]` to check the type or model for a host.

CHAPTER

5

Managing Users

User accounts

Your cluster uses a number of accounts to run properly.

In a mixed cluster, you need one OS account for each platform of the OS accounts.

OS account	Description	Used to ...s
Installation OS account	The OS account used when installing on a host.	Not usually used again once installation is complete.
System services execution user OS account	The OS account that runs most of the Platform agents, daemons, and services on a host.	This is the account that runs all the elements of the infrastructure of the cluster, including: • LIM • PEM • EGOSC (EGO service controller) • WEBGUI • Web Service Gateway (WSG) • SD • RS (repository service) • DerbyDB (if applicable) • PLC • Purger • Service Director
Application workload execution user OS account	The OS account that runs work. Requires Platform Symphony or another workload component.	Depending on installation and configuration decisions, there could be multiple application workload execution user OS accounts.
Cluster administrator OS account	The OS account that has permission to change cluster configuration and control the cluster.	On Windows management hosts, system services execute with this OS account. • VEMKD • SSM

There are also Platform user accounts used in your cluster as well. See more information in the following table.

Platform account	Description	Used to ...
EGO authentication users	An EGO authentication user account is a Platform system user who can be assigned to any role: cluster administrator, consumer administrator, or consumer user. Once you have created user accounts, you can assign roles to them in the Platform Management Console. **Restriction:** Only cluster administrators have access to **User Accounts** in the Platform Management Console.	The Platform user accounts used to log on to the Platform Management Console and the `egosh` command line interface (`egosh user logon`).

Allowable characters

A user account name can have the following characters: any alphanumeric character (a-z, 0-9), dashes (-), and underscores (_).They are case sensitive.

The length limitation of password is 16 and the allowed characters are:

!#$%&'()+,-./0123456789;=@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]
^_`abcdefghijklmnopqrstuvwxyz{}~

Not supported: "<>?|*: and characters from languages other than English.

User roles

There are three user roles that can be assigned to any user account. Each user role has a specific level of accessibility and control in the Platform Management Console.

Role	Description
Cluster Administrator	A "super user" able to accomplish all administrative and workload tasks, with access to all areas of the Platform Management Console and all actions within it. Entry to the Platform Management Console is through the dashboard, a heads up display that gives you a running summary of the health of your cluster.
Consumer Administrator	Access and control only over own branch of the tree. Consumer administrators are assigned at the top-level consumer and they are administrators for all sub-consumers in that branch of the tree. Entry to the Platform Management Console is through the dashboard, a heads up display that gives you a running summary of the health of your cluster.
Consumer User	Access and control over their own workload units only. Consumer users are assigned to individual consumers on your tree.

You can assign one user role to different user accounts for different consumers.

Add a cluster administrator

You must be a cluster administrator to create a user account or change cluster settings.

1. Navigate to Cluster > Summary.

   Cluster information displays in the Summary page.
2. Click Cluster Properties.

   Current cluster settings display in the Cluster Properties page.
3. Select the user account you want to have the cluster administrator role and click Add.
4. Click Apply.

Remove a cluster administrator

You must be a cluster administrator to delete a user account.

To remove a cluster administrator, you can delete their user account. Deleting a user account removes it from any role it may have been assigned everywhere in your consumer tree, including the cluster.

If you do not want to delete the user account, you can remove administrator privileges for that account in the cluster properties.

1. Click Cluster > Summary > Cluster Properties.

   In the second section, user accounts with cluster administrator privileges are on the right.
2. Use the Add and Remove buttons to modify the user accounts assigned as cluster administrators.
3. Click Apply.

Create a user account

You must be a cluster administrator to create a user account.

Create user accounts for each user you want to assign to a role for consumers. One user account can be assigned to multiple consumers with multiple roles. If the same user account is assigned more than one role to the same consumer, the highest role is assumed. Once you have created a user account, you can assign it a role when you create or modify a consumer.

1. Click Cluster > Configure User Accounts.

   A list of existing user accounts displays.
2. From Global Actions, select Create New User Account.

   The Create a New User Account page opens.
3. Fill in the fields.
   - The user name and password are mandatory.
   - The user name must be unique.
4. Click Create.

   Your new user account is displayed in the list.

You must assign this user account a role. You can assign it as a cluster administrator from Cluster > Summary > Cluster Properties or as a consumer user or consumer administrator in any consumer properties.

Modify a user account

You must be a cluster administrator to modify a user account.

1. Click Cluster > Configure User Accounts.
2. Click the user account name you want to modify.

   The User Account Properties page opens.
3. Make any changes.

   You cannot change the user name.
4. Click Apply.

   Your modified user account displays in the list.

To change the roles that this user account has been assigned, you need to modify the consumer properties for each consumer where this user account has been assigned a role.

Delete a user account

You must be a cluster administrator to delete a user account.

User accounts can be deleted even if they are assigned a role in the cluster or for any consumer. A deleted user account is automatically removed from all consumers where it is assigned.

You cannot delete the default cluster administrator user account (Admin).

1. Click Cluster > Configure User Accounts.

   Any existing user accounts are listed.
2. Locate the user account you want to delete and click Actions > Delete User Account.
3. Confirm that you want to delete the user account.

Add a consumer administrator

You must create at least one consumer before you can add a consumer administrator role to it. You can also add a consumer administrator during the creation of your first consumer.

Consumer administrators are mid-level administrators that may not have access to the entire cluster but only to specific branches of the consumer tree. They administer performance and planning for specific consumers and all consumers in the same branch. A consumer administrator is a user account set as a consumer administrator during the creation or modification of a consumer.

Note:

You can only add or remove a consumer administrator at the top-level consumer. The user account you specify as a consumer administrator is automatically an administrator for all consumers in that branch.

1. Create a new user account.
2. Go to Consumers > Consumer & Plans.

   Existing consumers display.
3. Select the top-level consumer you want to assign a consumer administrator to. Alternatively, you can create a new consumer from the Global Actions list.

   Depending on your action, either the Consumer Properties or Create a Consumer page displays.
4. From the section Specify administrators for this consumer, select the user account from the list and click Add.

   The user account you selected is added to the list of administrators for this consumer.

   You can specify as many administrators as you want for a consumer. Move multiple user accounts to the administrator list at once by holding down either Shift or Ctrl while you click.
5. Click Apply (or Create) to save your changes.

   Your page closes and the list of consumers and their properties refreshes.

You have added a consumer administrator to a top-level consumer and any current or future consumers in this branch.

Add a consumer user

You must create at least one consumer before you can add a user to it. You can add a user during the creation of your first consumer. To add a consumer user to a consumer, you must be a consumer administrator assigned to this consumer or a cluster administrator.

Consumer users are assigned to all consumers. A consumer user only has access to those consumers they have been assigned to. A consumer user is a user account set as a consumer user during the creation or modification of a consumer.

1. Create a new user account.
2. Navigate to Consumers > Consumer & Plans.

   Any existing consumers display.
3. Select the consumer you want to assign a consumer user to. Alternatively, you can create a new consumer from the Global Actions list.

   Depending on your action, either the Consumer Properties or Create a Consumer page displays.
4. From the section Specify users for this consumer, select the user account from the list and click Add.

   The user account you selected is added to the list of users for this consumer.
5. Click Apply (or Create) to save your changes.

   Your page closes and the list of consumers and their properties refreshes.

CHAPTER

6

Monitoring Resource Allocation

About resource allocation monitoring

Resource allocation monitoring, available only in the Platform Management Console, allows you to view your current owned, guaranteed, and shared resource allocation usage.

Resource allocation information

Data is current

The information shown is the most current data.

For historical analysis of resource allocation patterns over time, use the Reporting feature.

Host data is aggregated

The information shown combines data from all hosts. If a consumer is configured to use multiple resource groups, you cannot know how many of the consumer's slots are from each resource group.

Bar chart with a logarithmic scale

Allocation and demand is measured by number of slots. The information is displayed using a bar chart with a logarithmic scale that can indicate both small and large values. The scale reaches to 100,000 slots, regardless of cluster size or workload volume (the scale can indicate when a value exceeds 100,000 but the values can no longer be compared graphically).

Because the scale is not proportional, you should always roll over the chart and check the numeric values to best judge the allocation to each consumer.

Data filtering options

Depending on your focus in the tree, you can view resource allocation information for the entire cluster, one branch, or a single consumer.

List sorting options

By default, the list shows consumers organized hierarchically. You can sort by other criteria, such as Current Demand. The default then is to sort in ascending order, but you can reverse the sorting order by repeating the selection of the sort criteria.

Data available per consumer

For each consumer, you can view the following:

Allocation and Shortfall	Allocation is the number of slots allocated to the consumer. Shortfall occurs when a consumer's currently allocated slots (in non-management resource groups) are fewer than the number of guaranteed slots and there is an unsatisfied demand on those resource groups. This could occur if you have too few slots available in the hosts belonging to your resource group and that resource group has too many consumers drawing from it.
Owned, Shared, and Borrowed	The allocation is subdivided into the number of owned slots allocated to the consumer, the number of slots allocated to the consumer from the shared pool, and the number of slots borrowed by the consumer.

	When you roll over different colored sections of the bar chart, you see the numeric values for owned, shared, and borrowed.
Total Demand and Unsatisfied Demand	Total demand is the total number of slots requested by the consumer. Some or all of these slots may be allocated by EGO. Unsatisfied demand is the number of additional slots needed by the consumer to satisfy all demand. It is the difference between the number of slots requested and the number of slots allocated. When you roll over different colored sections of the bar chart, you see the numeric value for demand and unsatisfied demand.
Guaranteed	Guaranteed slots are the minimum number of slots to be allocated to the consumer if the consumer has demand. The guaranteed number of slots depends on the resource plan. The calculation adds the number of owned slots to the number of shared slots that the consumer should get if all consumers had maximum demand.
Cluster Size	Cluster size is the total number of slots in the cluster. This is shown for comparison purposes on each bar chart.

Relative allocation data

Optionally, you can see the current allocation in a pie chart, which shows the proportional distribution of slots at a glance. The legend identifies the 5 consumers that have the greatest allocation. Roll over the chart to see details for each consumer. This chart does not compare the current allocation to the resource plan.

For example, at the cluster level, you can see how many slots are allocated to each consumer, and how many slots are unallocated.

You can view relative allocation at the cluster level or at any branch of the consumer tree.

View resource allocation information

The bar chart shows detailed information for each consumer using a logarithmic scale that allows you to see details of each consumer.

Use sorting to help you compare consumers. For example, sort by Outstanding Demand to compare consumers most in need of additional resources with consumers least in need (top and bottom of list).

1. In the Console, navigate to Resources > Monitor Resource Allocation.
2. In the tree, select the focus (entire cluster or tree branch).

   A list of consumers appears, showing resource allocation information for each.

Compare relative resource allocation

The relative allocation chart shows consumers in a pie chart, to quickly judge the proportional allocation of resources among consumers.

1. In the Console, navigate to Resources > Monitor Resource Allocation.
2. In the tree, select the focus (entire cluster or tree branch).

   A list of consumers appears.
3. Click Show Relative Allocation.

   A pie chart shows the relative allocation of hosts to the consumers. The legend identifies up to five consumers.

View resource allocation shortfall

Shortfall occurs when a consumer's currently allocated slots (in non-management resource groups) are fewer than the number of guaranteed slots and there is an unsatisfied demand on those resource groups. This could occur if you have too few slots available in the hosts belonging to your resource group and that resource group has too many consumers drawing from it. You should add more member hosts to an under-allocated resource group or assign more resource groups to the consumers involved.

You can quickly identify if this condition exists in either the cluster or in one branch of the tree.

1. In the Console, navigate to Resources > Monitor Resource Allocation.
2. In the tree, select the focus (entire cluster or tree branch).

   A list of consumers displays.
3. Sort by Under Allocation and repeat the sort selection to list consumers in descending order.

   If any consumer is under allocated, the Current Allocation is highlighted and the details of the under allocation are displayed.

View resource allocation details for a consumer

You may view details such as the number of slots borrowed by a consumer.

1. In the Console, navigate to Resources > Monitor Resource Allocation.
2. In the tree, select the focus (entire cluster or tree branch).

   A list of consumers displays.
3. For each consumer, roll over different colored sections of the bar chart to see numeric values for owned, shared, and borrowed.

CHAPTER

7

Reporting

If you are using LSF, refer to the Reports chapter of the LSF Administrator's Guide instead.

About producing reports

The reports stored in the system do not include actual data. Instead, the reports define what data to extract from the system, and how to display it graphically.

Reports need to be produced before you can see the data. When you produce a report, you query the database and extract specific data. The amount of system overhead depends on how much data is in the report.

Reports have configurable parameters so you can modify the report and get exactly the data that you want.

About exporting reports

Data expires from the database periodically, so producing a report at a later date may return different data, or return no output at all. After you produce a report, you can keep your results by exporting the report data as comma-separated values in a CSV file. In this way you can preserve your data outside the system and integrate it with external programs, such as a spreadsheet. You can also keep your graphical results by using your browser to save the report results as an image.

Standard reports

For your convenience, Platform provides several standard reports for you to use. These reports allow you to keep track of some useful statistics in your cluster.

Standard reports are based on raw data stored in the relational database, and do not perform any data aggregation or calculations.

The following is a list of the standard reports that are included with the reporting feature. For further details on a report, view the full description of a standard report using the Console.

Note:

Depending on what you have installed, not all reports are available.

Name	Description	Category
Active Job States Statistics by Queue	Number of active jobs in each active job state in a selected queue.	LSF
Cluster Availability - EGO	EGO host availability in a cluster.	EGO
Cluster Availability - LSF	LSF host availability in an LSF cluster.	LSF
Cluster Job Hourly Throughput	Number of submitted, exited, and done jobs in a cluster.	LSF
Cluster Job Slot Utilization	Job slot utilization levels in your cluster.	LSF
Cluster Slot Utilization - EGO	Percentage of total slots used in the cluster, averaged hourly.	EGO
Desktop Utilization	Desktop utilization at each MED host or the entire cluster. You can only produce this report if you use LSF Desktop.	LSF Desktop
Host Resource Usage	Resource usage trends for selected hosts.	EGO
Hourly Desktop Job Throughput	Number of downloaded and completed jobs for each MED host or the entire cluster. You can only produce this report if you use LSF Desktop.	LSF Desktop
Job Slot Usage by Application Tag	Job slots used by applications as indicated by the application tag.	LSF
Jobs Forwarded to Other Clusters	The number of jobs forwarded from your cluster to other clusters. You can only produce this report if you use LSF MultiCluster.	LSF MultiCluster
Jobs Received from Other Clusters	The number of jobs forwarded to your cluster from other clusters. You can only produce this report if you use LSF MultiCluster.	LSF MultiCluster
License Usage	The license usage under License Scheduler. You can only produce this report if you use LSF License Scheduler.	LSF License Scheduler
Resource Allocation v. Resource Plan	Actual resource allocation compared to resource plan and unsatisfied resource demand for the selected consumer.	EGO

Name	**Description**	**Category**
Service Level Agreement (SLA)	Job statistics by job state over time, compared with SLA goals.	LSF

Custom reports

You can create and use custom reports if the standard reports are insufficient for your needs.

While standard reports are provided for your use by Platform, custom reports are reports you create as needed to satisfy specific reporting needs at your site.

Custom reports let you define combinations of data that are not available in the standard reports. Custom report output is always displayed in tabular format.

Create custom reports

The easiest way to create a custom report is to copy an existing report, then customize the SQL query string as desired. To customize the SQL query string, you may need to refer to the data schema, which describes the organization of information in the relational database. The data schema for each standard report is available in the Console by opening the report and clicking Help.

Even if you cannot edit SQL, saving a report as a custom report lets you re-use the report data without having to re-input the parameters in the standard report.

- If the time period is fixed, you get the same data every time you produce the report, but the report will be empty when the data expires from the database.
- If the time period is relative, you can get data for a different time period each time you produce the report.

You can also define custom reports from a blank template and input the SQL query string directly.

When you create custom reports, you can enter a category and use it to group the reports any way you want.

Delete custom reports

Unlike standard reports, custom reports can be deleted. You might prefer to rename old reports (by modifying them) instead of deleting them.

Reports directory

The reporting feature resides in the `perf` directory, which is a subdirectory of the top-level EGO directory.

If you ran `egoconfig mghost` to configure management hosts, the top-level EGO directory refers to the top-level EGO shared directory (for example, this may be `/share/ego` in Linux or `\\HostF\EGOshare` in Windows). If you did not configure management hosts, the top-level EGO directory refers to the EGO installation directory (by default, this is `/opt/ego` in Linux, or `C:\EGO` in Windows).

This document uses *`EGO_TOP`* to describe the top-level EGO directory.

Log files

Log files for the reporting services and data loaders are available in the `logs` subdirectory in the PERF directory (*`EGO_TOP`*`/perf/logs`).

There are seven logging levels that determine the detail of messages recorded in the log files. In decreasing level of detail, these are ALL (all messages), DEBUG, INFO, WARN, ERROR, FATAL, and OFF (no messages). By default, all service log files log messages of INFO level or higher (that is, all INFO, WARN, ERROR, and FATAL messages).

You can change the logging level of the reporting services or the data loaders by editing the `log4j.properties` file.

You can dynamically change the logging level of the `plc` service or the data loaders using the loader controller client tool, but these changes will be lost if you restart the `plc` service.

Event data files

The events logger stores event data in event data files.

The EGO allocation event data file is named `ego.stream` by default and has a default maximum size of 10MB.

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. The event data loaders read both the data files and the archive files.

The default archive file name is `ego.stream.0` for EGO, and the data and archive files are both located in `EGO_TOP/kernel/work/data` by default.

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.

You can manage your event data files by editing the relevant system configuration files. Edit `ego.conf` for the EGO allocation event data file configuration.

CHAPTER

8

Reports: Data Sources

If you are using LSF, refer to the Reports chapter of the LSF Administrator's Guide instead.

Data sources

Data sources provide the cluster operation data to the data loaders. Data sources include daemon status files, log files, and event files containing cluster operation data.

Configuration of the data sources

After editing the `plc_service.xml` or `ego.conf` configuration files, restart EGO on the master host (`egosh ego restart` *master_host_name*) for your changes to take effect.

Action	Configuration files	Parameter and syntax
Change the size of the EGO allocation event data files.	`ego.conf`	`EGO_DATA_MAXSIZE =` *max_alloc_file_size* where - *max_file_size* is the maximum size of the allocation event data file before the events logger creates an archive.
Change the location of the EGO allocation event data files.	`ego.conf`	`EGO_DATA_FILE =` *alloc_file_path* where - *alloc_file_path* is the path to the allocation event data file. This includes the name of the allocation event file.

Default behavior of data sources

The EGO allocation event data file is named `ego.stream` and has a maximum size of 10 MB. When a data file exceeds this size, the events logger archives the file and creates a new data file. The events logger only maintains one archive file and overwrites the old archive with the new archive. The archive file is named `ego.stream.0`. The two EGO files are located in *`EGO_TOP`*`/kernel/work/data`.

Data source interactions

Sampling data loaders request cluster operation data from the data sources while other data loaders obtain it directly. The data loaders store this data into tables within the relational database containing raw data. Each data loader contains data that is stored in specific tables in the raw database.

CHAPTER

9

Reports: Loader Controller

If you are using LSF, refer to the Reports chapter of the LSF Administrator's Guide instead.

Loader controller

The Platform loader controller (`plc`) manages the data loaders by controlling the schedule in which they gather data from the system.

Configuration of the loader controller

After editing the loader controller service configuration file (`plc_service.xml`), restart the `plc` service and EGO on the master host for your changes to take effect.

Action	Configuration files	Parameter and syntax
Enable automatic startup of the `plc` service. This is the default behavior.	`plc_service.xml` File location: *`EGO_TOP`*`/eservice/esc/conf/services`	`<sc:StartType>AUTOMATIC</sc:StartType>`
Disable automatic startup of the `plc` service.		`<sc:StartType>MANUAL</sc:StartType>`
Specify the default log level of your `plc` log file.	`log4j.properties` File location: *`EGO_TOP`*`/perf/conf`	`log4j.logger.`*domain_name*`.perf.dataloader=`*log_level*`,` *domain_name*`.perf.dataloader` where • *log_level* is the default log level of your loader controller log files. The loader controller only logs messages of the same or lower level of detail as *log_level*. Therefore, if you change the log level to `ERROR`, the loader controller will only log `ERROR` and `FATAL` messages.

Default behavior of the loader controller

The loader controller service starts automatically when the master host starts up.

Loader controller interactions

The loader controller service controls the scheduling of the data loaders. Sampling data loaders request cluster operation data from the data sources while other data loaders obtain it directly. The data loaders store this data into tables within the relational database containing raw data. Each data loader contains data that is stored in specific tables in the raw database.

CHAPTER

10

Reports: Data Loaders

If you are using LSF, refer to the Reports chapter of the LSF Administrator's Guide instead.

Data loaders

Data loaders gather cluster operation data and load it into tables in a relational database containing raw data. Data loaders are controlled by the Platform loader controller (`plc`) service.

Data loaders are polling loaders or history data loaders. The data loaders gather data and load this data into specific tables in the relational database containing raw data. Data loaders handle daylight savings automatically by using GMT time when gathering data.

Configuration of the data loaders

After editing the loader controller configuration files, restart the `plc` service for your changes to take effect.

Action	**Configuration files**	**Parameter and syntax**
Specify the frequency of data gathering for the specified data loader.	Loader controller configuration files for your data loaders: • `plc_ego.xml` (EGO) • `plc_soam.xml` (Symphony) File location: *`EGO_TOP`*`/perf/conf/plc`	`<DataLoader Name="`*loader_name*`" Interval="`*gather_interval*`" ... />` where • *loader_name* is the name of your data loader • *gather_interval* is the time interval between data gathering, in seconds
Enable data gathering for the specified data loader. This is the default behavior.		`<DataLoader Name="`*loader_name*`" ... Enable="true" ... />` where • *loader_name* is the name of your data loader
Disable data gathering for the specified data loader.		`<DataLoader Name="`*loader_name*`" ... Enable="false" ... />` where • *loader_name* is the name of your data loader
Enable data loss protection for the specified data loader. This is the default behavior.	Specific data loader configuration file: *dataloader_name*`.xml` File location: *`EGO_TOP`*`/perf/conf/dataloader`	`<Writer ... EnableRecover="Y">`
Disable data loss protection for the specified data loader.		`<Writer ... EnableRecover="N">`
Specify the default log level of your data loader log files.	`log4j.properties` File location: *`EGO_TOP`*`/perf/conf`	`log4j.logger.`*`domain_name`*`.perf.dataloader=`*`log_level`*`,` *`domain_name`*`.perf.dataloader` where • *log_level* is the default log level of your data loader log files. The data loaders only log messages of the same or lower level of detail as *log_level*. Therefore, if you change the log level to `ERROR`, the data loaders will only log `ERROR` and `FATAL` messages.

Data gathering methods

The data loaders use different methods of gathering data, depending on the types of data sources from which the data loaders gather data.

Collect	A data collecting loader has full control over what data is gathered from the data sources.
Retrieve	A data retrieving loader does not have full control over what data is gathered and needs to send a request to the data sources. The data sources send the requested set of logs or events back to the data loader.
Sample	A data sampling loader does not have full control over what data is gathered and needs to send a request to the data sources. The data sources send the requested system status information back to the data loader.

Default behavior of data loaders

Data loaders gather data from data sources at regular intervals. The following are lists of the data loaders and default behavior:

Table 1: EGO data loaders

Data loader name	Data type	Data gathering interval	Data loads to	Loader type	Data gathering method

Consumer resource (`egoconsumerresloader`)	resource allocation	5 minutes	`CONSUMER_DEMAND`
			`CONSUMER_RESOURCE_ALLOCATION`
			`CONSUMER_RESOURCELIST`
				polling	sample

Dynamic metric (`egodynamicresloader`)	host-related dynamic metric	5 minutes	`RESOURCE_METRICS` `RESOURCES_RESOURCE_METRICS`	polling	sample
EGO allocation events (`egoeventsloader`)	resource allocation	5 minutes	`ALLOCATION_EVENT`	polling	collect
Static attribute (`egostaticresloader`)	host-related static attribute	1 hour	`ATTRIBUTES_RESOURCE_METRICS` `RESOURCE_ATTRIBUTES`	polling	sample

Table 2: Symphony data loaders

Data loader name	Data type	Data gathering interval	Data loads to	Loader type	Data gathering method
Session (`symsessionloader`)	open and suspended sessions	5 minutes	`SESSION_ATTRIBUTES`	polling	sample
Session history (`symsessionhistloader`)	switched session history	5 minutes	`SESSION_HISTORY`	history	collect
Task history (`symtaskhistloader`)	switched task history	5 minutes	`TASK_ATTRIBUTES`	history	collect

- The session data loader will miss the session data if the session started and finished within the sampling interval.

- The task history data loader will start to collect data within three days before the first start time of the loader controller. The data loader will continue to collect data as long as the loader controller is running.

Data loader interactions

The loader controller service controls the scheduling of the data loaders. Sampling data loaders request cluster operation data from the data sources while other data loaders obtain it directly. The data loaders store this data into tables within the relational database containing raw data. Each data loader contains data that is stored in specific tables in the raw database.

CHAPTER

11

Reports: Data Purger

If you are using LSF, refer to the Reports chapter of the LSF Administrator's Guide instead.

Data purger

The data purger (`purger`) service maintains the size of the database by archiving old records and purging them from the database.

Configuration of the data purger

After editing the data purger configuration files, restart the `purger` service for your changes to take effect.

After editing the data purger service configuration file (`purger_service.xml`), restart the `purger` service and EGO on the master host for your changes to take effect.

Action	**Configuration files**	**Parameter and syntax**
Specify the default duration of time that the records are stored in the database before being purged.	`purger` configuration files: • `purger_ego.xml` (EGO) • `purger_soam.xml` (Symphony) File location: *`EGO_TOP/`*`perf/conf/purger`	`<TableList Duration="`*expiry_time*`">` where • *expiry_time* is the expiry time, in days, up to a maximum of 31 days This default expiry time applies to all tables, but may be overridden for individual tables by specifying a record expiry time for that table.
Specify the duration of time that the records in a specific table are stored in the database before being purged.		`<Table TableName="`*table_name*`" ... Duration="`*expiry_time*`">` where • *table_name* is the name of the individual table • *expiry_time* is the expiry time, in days, up to a maximum of 31 days This expiry time overrides the default expiry time for all records as specified in the `<TableList>` element.
Specify the daily time in which the data purger purges old data.	`purger_service.xml` File location: *`EGO_TOP/`*`eservice/esc/conf/services`	`<ego:Command>...` *file_path*`/purger... -t` *purge_time* where • *purge_time* is the 24-hour daily time, in `hh:mm` format.
Specify the time interval in which the data purger purges old data, starting from when the `purger` service first starts up.		`<ego:Command>...` *file_path*`/purger... -t * [`*purge_time_interval*`]` where • *purge_time_interval* is the time interval, in hours.
Enable automatic startup of the `purger` service. This is the default behavior.		`<sc:StartType>AUTOMATIC</sc:StartType>`
Disable automatic startup of the `purger` service.		`<sc:StartType>MANUAL</sc:StartType>`

Action	**Configuration files**	**Parameter and syntax**
Specify the default log level of your `purger` log file.	`log4j.properties` File location: *`EGO_TOP/`*`perf/conf`	`log4j.logger.`*`domain_name`*`.perf.purger=`*`log_level`*`, `*`domain_name`*`.perf.purger` where - *log_level* is the default log level of your data purger log files. The data purger only logs messages of the same or lower level of detail as *log_level*. Therefore, if you change the log level to `ERROR`, the data purger will only log `ERROR` and `FATAL` messages.

Record archives

The data purger archives old data to the `EGO_TOP/perf/work/archive` directory. This file is a ZIP file containing a CSV file. The archive file names use the following format:

*tablename*_*date* *timestamp*`.zip`

where

- *tablename* is the name of the table from where the records are archived.
- *date* is the date of the record in `YYYY-MM-DD` format.
- *timestamp* is the 24-hour time stamp of the record in `HH-MM-SS` format.

Note:

There is a space between the date and the time stamp.

For example, `CONSUMER_DEMAND_2007-01-15 22-15-00.zip` is the name of an archive of the records from the `CONSUMER_DEMAND` table at 10:15:00 p.m. on 15 January, 2007.

Default behavior of the data purger

The data purger service starts automatically when the master host starts up. The data purger maintains records for 14 days before archiving and purging them at 12:30 a.m. every day. The data purger is a service that starts up automatically whenever EGO starts up.

Data purger interactions

The data purger moves old records from tables in the relational database and archives them.

CHAPTER

12

Managing Reports

If you are using LSF, refer to the Reports chapter of the LSF Administrator's Guide instead.

Disable automatic startup of the reporting services

1. Navigate to the EGO service directory.

   For example,

   Linux: **cd *EGO_TOP*/eservice/esc/conf/services**

   Windows: **cd *EGO_TOP*\eservice\esc\conf\services**
2. Edit the service configuration file and change the service type from automatic to manual.

   In the `<sc:StartType>` tag, change the text from AUTOMATIC to MANUAL.
3. Stop the service that you changed.
4. In the command console, restart EGO on the master host to activate these changes.

   egosh ego restart *master_host_name*

View the status of the loader controller

Use the loader controller client tool to view the status of the loader controller.

1. Launch the loader controller client tool with the `-s` option. The loader controller is located in `EGO_TOP/perf/version_number/bin`.
    - In UNIX, run **plcclient.sh -s**.
    - In Windows, run **plcclient.bat -s**.

Dynamically change the log level of your loader controller log file

Use the loader controller client tool to dynamically change the log level of your `plc` log file if it does not cover enough detail, or covers too much, to suit your needs.

If you restart the `plc` service, the log level of your `plc` log file will be set back to the default level. To retain your new log level, change the default level of your plc log file.

1. Launch the loader controller client tool with the `-l` option. The loader controller is located in `EGO_TOP/perf/version_number/bin`.
    - In UNIX, run **plcclient.sh -l** *log_level*.
    - In Windows, run **plcclient.bat -l** *log_level*.

    In decreasing level of detail, the log levels are ALL (for all messages), DEBUG, INFO, WARN, ERROR, FATAL, and OFF (for no messages).

Dynamically change the log level of your data loader log files

Use the loader controller client tool to dynamically change the log level of your individual data loader log files if they do not cover enough detail, or cover too much, to suit your needs.

If you restart the `plc` service, the log level of your data loader log files will be set back to the default level. To retain your new log level, change the default level of your data loader log files.

1. If you are using the default configuration file, launch the loader controller client tool with the `-n` and `-l` options. The loader controller is located in `EGO_TOP/perf/version_number/bin`.

   - In UNIX, run **plcclient.sh -n** *data_loader_name* **-l** *log_level*.
   - In Windows, run **plcclient.bat -n** *data_loader_name* **-l** *log_level*.

   In decreasing level of detail, the log levels are ALL (for all messages), DEBUG, INFO, WARN, ERROR, FATAL, and OFF (for no messages).

   For example, to change the consumer resource data loader log files to the ERROR log level:

   - In UNIX,

     ```
     plcclient.sh -n egoconsumerresloader -l ERROR
     ```
   - In Windows,

     ```
     plcclient.bat -n egoconsumerresloader -l ERROR
     ```

Change the default log level of your reporting log files

Change the default log level of your log files if they do not cover enough detail, or cover too much, to suit your needs.

1. Edit `EGO_TOP/perf/conf/log4j.properties`.
2. Navigate to the section representing the service you want to change, or to the default loader configuration if you want to change the log level of the data loaders, and look for the **.logger.** variable.

   For example, to change the log level of the data purger log files, navigate to the following section, which is set to the default INFO level:

   ```
   # Data purger ("purger") configuration
   log4j.logger.domain.perf.purger=INFO, domain.perf.purger
   ```
3. Change the **.logger.** variable to the new logging level.

   In decreasing level of detail, the valid values are ALL (for all messages), DEBUG, INFO, WARN, ERROR, FATAL, and OFF (for no messages). The services or data loaders only log messages of the same or lower level of detail as specified by the *.logger.* variable. Therefore, if you change the log level to ERROR, the service or data loaders will only log ERROR and FATAL messages.

   For example, to change the data purger log files to the ERROR log level:

   ```
   # Data purger ("purger") configuration
   log4j.logger.domain.perf.purger=ERROR, domain.perf.purger
   ```
4. Restart the service that you changed (or the `plc` service if you changed the data loader log level).

Change the disk usage of EGO allocation event data files

If your system logs a large number of events, increase the disk space allocated to the EGO allocation event data files. If your disk space is insufficient, decrease the space allocated to the EGO allocation event data files or move these files to another location.

1. Edit `ego.conf`.

   Windows: `%EGO_CONFDIR%`

   Linux: `$EGO_CONFDIR`

   a) To change the size of each EGO allocation event data file, specify or change the EGO_DATA_MAXSIZE parameter.

      `EGO_DATA_MAXSIZE =` *integer*

      If unspecified, this is `10` by default. Change this to the new desired file size in MB.

   b) To move the files to another location, specify or change the EGO_DATA_FILE parameter.

      `EGO_DATA_FILE =` *file_path*

      If unspecified, this is *`EGO_TOP`*`/kernel/work/data/ego.stream` by default.

2. In the command console, restart EGO on the master host to activate this change.

   egosh ego restart *master_host_name*

Change the data purger schedule

To reschedule the deletion of old data, change the time in which the data purger deletes the old data.

1. Edit `EGO_TOP/eservice/esc/conf/services/purger_service.xml`.
2. Navigate to `<ego:Command>` with the -t parameter in the purger script.
    - In UNIX, this is
      ```
      <ego:Command> ... .../purger.sh -t ...
      ```
    - In Windows, this is
      ```
      <ego:Command> ... ...\purger.bat -t ...
      ```

   By default, the data purger is scheduled to delete old data at 12:30am every day.
3. Change the -t parameter in the data purger script to the new time (-t *new_time*).

   You can change the data purger schedule to a specific daily time, or at regular time intervals, in hours, from when the `purger` service first starts up.

   For example, to change the schedule of the data purger:
    - To delete old data at 11:15pm every day:
      ```
      <ego:Command> ... .../purger... -t 23:15
      ```
    - To delete old data every 12 hours from when the `purger` service first starts up:
      ```
      <ego:Command> ... .../purger... -t *[12]
      ```
4. Stop the `purger` service.
5. In the command console, restart EGO on the master host to activate these changes.

   egosh ego restart *master_host_name*

Change the default record expiry time

To reduce or increase the number of records stored in the database, change the duration of time that a record is stored in the database before it is purged. This applies to all tables in the database unless you also specify the record expiry time in a particular table.

1. In the `<TableList>` tag, edit the Duration attribute to your desired time in days, up to a maximum of 31 days.

   For example, to have the records purged after 7 days:

   ```
   <TableList Duration="7">
   ```

   By default, the records are purged after 14 days.
2. Restart the `purger` service.

Change the record expiry time per table

To reduce or increase the number of records stored in the database for a particular table, change the duration of time that a record is stored in the database for this table before it is purged. The duration only applies to this particular table.

1. Navigate to the specific `<Table>` tag with the TableName attribute matching the table that you want to change.

   For example:

   ```
   <Table TableName="RESOURCE_METRICS" TimestampColumn="TIME_STAMP" ... />
   ```
2. Add or edit the Duration attribute with your desired time in days, up to a maximum of 31 days.

   For example, to have the records in this table purged after 10 days:

   ```
   <Table TableName="RESOURCE_METRICS" TimestampColumn="TIME_STAMP"
   Duration="10" ... />
   ```
3. Restart the `purger` service.

Change the frequency of data collection

To change how often the data loaders collect data, change the frequency of data collection per loader.

1. Navigate to the specific `<DataLoader>` tag with the Name attribute matching the data loader that you want to change.

   For example:

   ```
   <DataLoader Name="egodynamicresloader" Interval="300" ... />
   ```
2. Add or edit the Interval attribute with your desired time in seconds.

   For example, to have this plug-in collect data every 200 seconds:

   ```
   <DataLoader Name="egodynamicresloader" Interval="200" ... />
   ```
3. Restart the `plc` service.

Disable data collection for individual data loaders

To reduce unwanted data from being logged in the database, disable data collection for individual data loaders.

1. Navigate to the specific `<DataLoader>` tag with the Name attribute matching the data loader that you want to disable.

   For example:

   ```
   <DataLoader Name="egodynamicresloader" ... Enable="true" .../>
   ```

2. Edit the Enable attribute to "false".

   For example, to disable data collection for this plug-in:

   ```
   <DataLoader Name="egodynamicresloader" ... Enable="false" ... />
   ```

3. Restart the `plc` service.

Test the reporting feature

Verify that components of the reporting feature are functioning properly.

1. Check that the reporting services are running.
   a) If you are running the Derby demo database, check that the State of the `derbydb` service is STARTED.
2. Check that there are no error messages in the reporting logs.
   a) View the `*EGO_TOP*/perf/logs/plc.`*host_name*`.log` file.
   b) Verify that there are no ERROR messages and that, in the `DataLoader Statistics` section, there are data loader statistics messages for the data loaders in the last hour.

      You need to find statistics messages for the following data loaders:

      - `egoconsumerresloader`
      - `egodynamicresloader`
      - `egoeventsloader`
      - `egostaticresloader`

   c) View the data purger and data loader log files and verify that there are no ERROR messages in these files.

      You need to view the following log files:

      - *EGO_TOP*`/perf/logs/dataloader/egoconsumerresloader.`*host_name*`.log`
      - *EGO_TOP*`/perf/logs/dataloader/egodynamicresloader.`*host_name*`.log`
      - *EGO_TOP*`/perf/logs/dataloader/egoeventsloader.`*host_name*`.log`
      - *EGO_TOP*`/perf/logs/dataloader/egostaticresloader.`*host_name*`.log`
      - *EGO_TOP*`/perf/logs/purger.`*host_name*`.log`
3. Check the report output.
   a) Produce a standard report.
   b) Verify that the standard report produces a chart or table with data for your cluster.

If you were not able to verify that these components are functioning properly, identify the cause of these problems and correct them.

Disable the reporting feature

You must have root or cluster administrator access on the master host.

1. Disable the EGO allocation events data logging.
   a) Define or edit the EGO_DATA_ENABLE parameter in the `ego.conf` file to disable data logging.

      ```
      EGO_DATA_ENABLE = N
      ```

   b) In the command console, restart EGO on the master host to activate these changes.

      egosh ego restart *master_host_name*
2. Stop the reporting services.
3. Disable automatic startup of the reporting services.

Move to a production database for EGO

The commercial database must be properly configured and running:

- You have a user name, password, and URL to access the database server.
- Your database server account has access to create triggers, sequences, tables, and stored procedures.
- There is appropriate space in the database allocated for the reporting feature.
- You installed the latest JDBC driver for the commercial database.

  The JDBC driver for an Oracle database (`ojdbc14.jar` or newer) is available from the following URL:

  http://www.oracle.com/technology/software/tech/java/sqlj_jdbc/index.html

The Derby demo database is not supported for any production clusters. To produce regular reports for a production cluster, you must use a supported commercial database.

The reporting feature supports Oracle 9i and 10g databases for production clusters.

All data in the demo database will not be available in the production database. Some of your custom reports may not be compatible with the production database if you used non-standard SQL code.

1. Create an Oracle database schema for your commercial database.
   a) Navigate to the database schema directory.

      For example,

      Linux: **cd *EGO_TOP*/perf/ego/*version_number*/DBschema/Oracle**

      Windows: **cd *EGO_TOP*\perf\ego\*version_number*\DBschema\Oracle**
   b) Run the scripts to create a database schema.

      sqlplus *user_name*/*password@connect_string* **@egodata.sql** *data_tablespace index_tablespace*

      where

      - *user_name* is the user name on the database server.
      - *password* is the password for this user name on the database server.
      - *connect_string* is the named SQLNet connection for this database.
      - *data_tablespace* is the name of the tablespace where you intend to store the table schema.
      - *index_tablespace* is the name of the tablespace where you intend to store the index.
2. Stop the reporting services.

   Stop the `derbydb` (if you are using the Derby demo database), `plc`, and `purger` services.
3. If you are using the Derby demo database, disable automatic startup of the `derbydb` service.
4. Copy the Oracle JDBC driver into the PERF and GUI library directories.

   You need to copy the Oracle JDBC driver to the following directories:

   - Linux:
     - `EGO_TOP/perf/version_number/lib`

    - *EGO_TOP*/gui/*version_number*/tomcat/common/lib
- Windows:
    - *EGO_TOP*\perf\version_number\lib
    - *EGO_TOP*\gui\version_number\tomcat\common\lib

5. Configure your database connection.
    a) Launch the database configuration tool.
        - In UNIX (X-Windows only), run ***EGO_TOP*/perf/*version_number*/bin/dbconfig.sh**.
        - In Windows, run ***EGO_TOP*\perf\version_number\bin\dbconfig.bat**.

    b) In the User ID and Password fields, specify the user account name and password with which to connect to the database and to create your database tablespaces.

    Note:

    This user account must have been defined in your database application, and must have read and write access to the database tables. In general, this is the name you used to create your Oracle tablespaces.

    c) In the JDBC driver field, select the driver for your commercial database.

    d) In the JDBC URL field, enter the URL for your database.

    This should be similar to the format given in Example URL format.

    e) In the Maximum connections field, specify the maximum allowed number of concurrent connections to the database server.

    This is the maximum number of users who can produce reports at the same time.
6. Restart the reporting services.
7. Restart the Platform Management Console.

    Note:

    The Platform Management Console will be unavailable during this step.

    a) In the command console, restart the `WEBGUI` service.

    egosh service stop WEBGUI

    egosh service start WEBGUI

The report data will now be loaded into the production database and the Console will use the data in this database.

PART

II

Managing Resource Distribution

CHAPTER

13

Managing the Tree

About the consumer tree

Overview

The consumer tree organizes consumers into a structure that makes it easy to apply resource plans.

The consumer tree is closely related to the resource plan. The plan cannot be defined without the tree.

The tree only defines organizational relationships among consumers, while the plan defines resource allocation.

The choice of consumers and their hierarchy should reflect long-term business goals because it can be complicated to modify the tree. To make the system adjust to short-term business changes, you can modify the users associated with a consumer, or the resource plans defined in the plan.

Terminology

Component	Description
Tree	The resource distribution tree identifies consumers of the cluster resources, and organizes them into a manageable structure.
Plan (Resource plan)	The resource plan describes the relationship between the consumer tree and resource groups, and defines plans for how cluster resources are to be shared among consumers.
Consumers	A consumer in the tree represents any entity that can demand resources from the cluster. A consumer might be a business service, a business process that is a complex collection of business services, an individual user, or an entire line of business. The consumers ManagementServices, SampleApplications, and ClusterServices, along with their sub-consumers, are installed by default. • ManagementServices has one sub-consumer, EGOManagementServices, which runs important system services on management hosts in the cluster. Services include derbydb, plc, purger, ServiceDirector, WEBGUI, RS, and WebServiceGateway. ManagementServices is configured to use the ManagementHosts resource group. Do not modify or delete this consumer. • SampleApplications has one sub-consumer, EclipseSamples. The SampleApplications consumer and its sub-consumer can be modified or deleted (although you want to use the provided EclipseSamples to begin using EGO right away—this is a sample consumer with registered applications that are ready to run). • ClusterServices is configured to use the InternalResourceGroup resource group. It has one sub-consumer, EGOClusterServices, which runs important system services on every host in the cluster. Do not modify or delete ClusterServices, or use it to run workload units.

Component	Description
Multi-level tree	Company projects are generally structured with multiple layers and components. For example, a project belongs to a department, a department to a business unit and so on. A multi-level consumer tree allows you to configure consumers in a hierarchical fashion to match your business structure. **Note:** As a best practice, restrict the number of tree levels to four.
Tree root	The root of the tree represents the entire cluster and all resources in it. Resources from the root are distributed through the tree to consumers.
Top-level consumers	The consumers attached directly to the root are called top-level consumers. The top-level consumer is the head of a consumer branch.
Leaf consumers	If a consumer has no descendants, it is called a leaf consumer. Services and applications can only be associated with leaf consumers. Borrow and lend policies are set at this level. **Note:** As a best practice, limit leaf consumers to fewer than 20.
Branches, descendants	If a consumer in the tree has other descendants, thereby creating a branch in a multi-level tree, it is called a branch consumer. Branch consumers exist to redistribute resources down the branch to their descendants. Descendants of a branch consumer may also have descendants, thereby becoming branch consumers themselves. Every branch in the tree ends with a leaf consumer.
Parent	A consumer containing another consumer (a "child"). A parent can contain a child consumer, or be the child of another parent consumer.
Sibling	Two or more consumers sharing the same parent consumer.
Child, sub-consumer	A consumer nested within another consumer (a "parent"). A child (or sub-consumer) of one parent can be the parent to another nested child. A leaf consumer is always a child at the end of the branch.
Consumer administrator	For ease of management, you can create consumer administrators for top-level consumers in a multi-level tree. These users can change the plan for lower-level consumers on their branch (descendants), without requiring cluster administrator permissions. Only a cluster administrator can change the plan for top-level consumers.
ClusterServices consumer	The special ClusterServices consumer is a built-in top-level consumer that uses the built-in InternalResourceGroup resource group and runs necessary EGO components. It contains one descendant, EGOClusterServices, and is allocated a certain number of slots by default. Do not modify this consumer or use it to run workload units.
ManagementServices consumer	The special ManagementServices consumer is a built-in top-level consumer that uses the built-in ManagementHosts resource group and runs necessary EGO components. It contains one descendant, EGOManagementServices. Do not modify this consumer or use it to run workload units.

Component	Description
SampleApplications consumer	This is an out-of-box sample consumer with registered applications that are ready to run (if you choose to install and run the SDK samples). It contains one descendant, EclipseSamples.

Consumer properties

Property	**Explanation**
Name	A unique name of a department or project in your structure.
Administrators	User accounts or with administrative privileges for this consumer and all consumers beneath it. A consumer administrator can see everything in their branch of the tree, modify any property, assign resources, modify user accounts, and control hosts.
Users	List of user accounts that have access to this consumer without administrative privileges. Users can only view their own properties.
OS user account	The operating system user account under which workload runs. All workload for the consumer runs under the same account, no matter which user submitted workload units. If the consumer runs workload units on a Windows host, include the domain name as shown: *domain_name*\user_name If the consumer runs workload units on a Linux host, EGO can automatically strip the domain name from the user name. For example, if you configure the account as mydomain\user2, it is interpreted as mydomain\user2 on Windows but as user2 on Linux. Every new Windows execution user account needs to have the password configured with `egosh ego execpasswd`. The egoadmin account is already configured. Any activity started through `egosh` uses the same execution account as configured for the leaf consumer it runs on. The file `ConsumerTrees.xml` contains execution user account information for each registered consumer.
Resource groups	A collection of hosts. You can create resource groups based on similar qualities or by machine names. Only those resource groups assigned to this consumer by its parent consumer are available for assignment.
Reclaim	Reclaim applies to consumers that are borrowing resources. If a client is running workload units on a borrowed resource, you can impose a delay (grace period) so that these units can run uninterrupted before the resource gets returned (reclaimed). Alternately, you can choose to immediately interrupt workload units running on a borrowed resource when it is reclaimed by setting the grace period to 1. When a grace period is left blank or set to 0, it defaults to 120 seconds.

Create a consumer

To create a consumer, you must either be a cluster administrator or a consumer administrator for the branch on which you are creating a consumer.

1. Click Consumers > Consumers & Plans.

   A list of existing top-level consumers in your tree displays.
2. Locate and click on the tree level for which you would like to add a consumer.
3. From Global Actions, select Create a Consumer.

   The Create a Consumer page displays.

   You can create a consumer at any level of your existing tree, except where a consumer already has something registered to it; in this case, no other consumers can be created below it in the tree.
4. Fill in the consumer properties.

   Some of these properties may be already filled out or disabled depending on which tree level you are adding a consumer to.
   a) Specify a name for your new consumer.

      The name can be up to 32 characters long and cannot be the same as any other consumer in your tree.
   b) If you have created new user accounts, choose one or more for this consumer; otherwise, leave this field blank.

      Specified administrators automatically become administrators for any other consumers created on this branch.
   c) Specify zero, one, or more users for this consumer.

      You must specify at least one user in order for the consumer to run workload.
   d) Specify the workload execution user account (the OS account under which workload runs).

      This field may be pre-populated, but you can modify it. Windows accounts should include a domain name.

      If you specify a Windows user account that has not already been configured, you have to log on to EGO as the cluster administrator and then run `egosh ego execpasswd` before the execution user can run an activity without exiting.
   e) Specify one or more resource groups this consumer should have access to.

      Only the resource groups specified by this consumer's parent are available for selection. If you have not modified your resource groups, you can keep the default resource group selections.
   f) Specify a reclaim grace period.

      The reclaim grace period is applied when a resource belonging to another consumer is now being reclaimed by its owner consumer. Setting the reclaim grace period high (compared to the average length of your workload) allows workload to finish before the resource is reclaimed. Setting the reclaim grace period to 1 terminates all workload running and reclaims the resource almost immediately.
   g) (Optional) Check the Rebalance when time intervals change box.

      If you want EGO to “rebalance” or reset the ownership when a new time interval occurs with a change in ownership of resources, check this box. Similarly, when resources are reclaimed (or passed back to their original owners), you can evoke a rebalancing in accordance with the resource plan.

Before EGO rebalances according to the resource plan, a consumer's grace period is honored to help ensure workload is completed before being killed.

5. Click Create.

You may need to configure the Windows password of the execution user account.

Configure Windows password

If, when creating or modifying a consumer, you specify a Windows user account that has not already been configured, you have to set the password before you can continue.

Configure the password of every new Windows execution account that you introduce into the system. (The egoadmin account was configured during installation.)

Use the same command to update the system if the execution account password ever changes. You have to register the new password in EGO every time the execution account password changes in Windows.

Restriction:

The password must be 31 characters or less.

---

1. Log on to a Windows host as egoadmin.
2. Log on to EGO as an EGO authentication user.

   egosh user logon -u *user_name* **-x** *password*

   For example, type

   ```
   egosh user logon -u Admin -x Admin
   ```
3. To configure the Windows password, run

   egosh ego execpasswd -u *user_name* -x *password*

   For example, if the account is mydomain\user2, type

   ```
   egosh ego execpasswd -u mydomain\user2 -x mypasswd
   ```

Delete a consumer

You cannot delete a consumer if it has anything registered to it or if any of its sub-consumers has anything registered to it. Unregister before deleting a consumer.

Deleting a consumer means that it and all of its sub-consumers no longer exist in your consumer tree. A deleted consumer does not have any allocations and cannot run workload.

1. Click Consumers > Consumers & Plans.

   A list of existing top-level consumers in your tree displays.
2. On your tree, click the consumer name above the one you want to delete.

   A list of consumers at this level displays.
3. Locate the consumer you want to delete in your list and select Actions > Delete.
4. Confirm that you want to delete the consumer.

View consumer properties

You must have created at least one consumer before you can view consumer properties.

You may need to see the properties you have assigned to a consumer.

Note:

You can also view properties in a secondary window by selecting the leaf consumer’s parent in the tree, and then clicking the name of the leaf consumer from the list.

1. Click Consumers > Consumers & Plans.

   A list of existing top-level consumers in your tree displays.
2. Locate and click on the top level or parent consumer your consumer belongs to.
3. Click Consumer Properties.

Modify consumer properties

You must have created at least one consumer before you can modify consumer properties.

You must either be a cluster administrator or a consumer administrator for this branch to modify consumer properties.

You can modify your consumer properties including user accounts or user groups who have access as consumer administrators and consumer users, execution user account, resources available for a consumer and all its sub-consumers, and reclaim behavior.

You can modify any consumer at any level of your existing tree, but do not modify the built-in ManagementServices consumer. You may see different fields for different levels of consumers in your tree.

1. Click Consumers > Consumers & Plans.

   A list of existing top-level consumers in your tree displays.
2. Locate the consumer you want to modify in the tree and click on it.

   A list of sub-consumers displays.
3. Click Consumer Properties.

   The Consumer Properties page displays.
4. Modify the consumer properties.

   The consumer name cannot be changed.

   a) Optional: Choose one or more administrators for this consumer. By default, the cluster administrator is the consumer administrator.

      This option is only available only for a top-level consumer (a consumer just beneath your cluster.)

   b) Choose zero, one, or more users for this consumer.

      If you do not specify at least one user, the consumer cannot be used to run workload.

   c) Specify the workload execution user account (the OS account under which workload runs).

      If you specify a Windows user account that has not already been configured, you have to run `egosh ego execpasswd` before you can use the consumer.

   d) Specify one or more resource groups that this consumer should have access to.

      If you do not specify at least one resource group, workload does not run for this consumer.

   e) Specify a reclaim grace period.

      The reclaim grace period is applied when a resource belonging to another consumer is now being reclaimed by its owner consumer. Setting the reclaim grace period high (compared to the average length of your workload) allows workload to finish before the resource is reclaimed. Setting the reclaim grace period to 1 kills all workload running and reclaims the resource almost immediately.

   f) (Optional) Check the Rebalance when time intervals change box.

      If you want EGO to “rebalance” or reset the ownership when a new time interval occurs with a change in ownership of resources, check this box. Similarly, when resources are reclaimed (or passed back to their original owners), you can evoke a rebalancing in accordance with the resource plan.

   Before EGO rebalances according to the resource plan, a consumer's grace period is honored to help ensure workload is completed before being killed.

5. Click Apply.

   The Consumer Properties page updates and your changes are saved.

You may need to configure the Windows password of the execution user account.

Rank consumers in order of priority

You must have created at least one consumer before you can set any priorities (rank). You must either be a cluster administrator or a consumer administrator for this branch to modify a consumer rank.

If you have critical workload to run, you can ensure resources are available by assigning a high rank to a consumer. Note that if all consumers have a high ranking, any advantage one may have over another is nullified. Be selective in assigning a high consumer rank to a consumer.

Consumer rank and the resource group or groups assigned to that consumer work in collaboration. Even if you set a consumer's rank high, the resource group must have the resources available. You can also enable borrowing from other consumers to make sure any unowned resources are assigned to your high ranking consumer.

1. Click Consumers > Consumers & Plans, and then Resource Plan.
2. Select Time Intervals and Settings > Show Advanced Settings.
3. Under ConsumerRank, rank as many consumers as you want.

   Specify any positive whole number, where 1 is the highest priority. Priority settings are relative to one another within the resource group.

   If you leave the priority blank, that consumer has no priority over any other consumer (it does not form part of any consumer ordering/sequencing).
4. When finished assigning priority settings, click Apply to save your changes.
5. From the drop down list of resource groups, switch resource groups until you have set consumer priorities for all resource groups within the consumer tree.

You may want to enable and specify details for lending and borrowing for this leaf consumer and its siblings, taking into consideration what priority you have set them.

For example, if you have set a low priority for a consumer, you may wish to enable lending with no limits for it, and then enable borrowing from this consumer in the borrowing details of all other consumers. Doing this maximizes the effectiveness of your resource distribution, lending and borrowing policies, and priority settings. In this example, low priority slots are dynamically lent out to higher priority consumers as required.

Create a plan for reclaiming a resource

You must have created at least one consumer before you can modify consumer properties. You must either be a cluster administrator or a consumer administrator for this branch to modify consumer properties.

Resources are reclaimed by resource owners with an unsatisfied demand. If you want to control how long workload runs on the borrowed resource once reclaimed, set a grace period. Specify this property when you first create a consumer, or modify it at any time. If you do not specify a grace period or specify one with the value 0, the default of 120 seconds is used.

1. Plan the ranking of your consumers knowing that resources are reclaimed based on their rank; those leaf consumers with a lower consumer rank are reclaimed before consumers with a higher rank.
   - Example 1: If a lending consumer has unsatisfied demand and requires that its lent resources be reclaimed, EGO looks to reclaim resources starting with leaf consumers with the lowest consumer rank.
   - Example 2: If a lending consumer has a specific resource requirement (for example, the lending consumer needs a Windows slot with a certain amount of available memory), EGO reclaims the first lent resource it finds that matches this requirement. Borrowing leaf consumers with the lowest consumer rank are considered first, followed by leaf consumers with a higher consumer rank.
2. Click Consumers & Plans.

   A list of any existing consumers at that level in your tree displays.
3. Create a new consumer, or locate and click the level on your tree that an existing consumer belongs to.

   You can modify any consumer at any level of your existing tree, but do not modify the built-in ManagementServices consumer. You may see different fields for different levels of consumers in your tree.
4. Click Consumer Properties.

   The Consumer Properties page displays.
5. Specify a reclaim grace period.

   The reclaim grace period is applied when a resource belonging to another consumer is now being reclaimed by its owner consumer. Setting the reclaim grace period high (compared to the average length of your workload) allows workload to finish before the resource is reclaimed. Setting the reclaim grace period to 1 terminates all workload running and reclaims the resource within one second.

   - Set the reclaim grace period high (compared to the average length of your workload) to allow workload to finish before the resource is reclaimed.
   - Set the reclaim grace period to 1 to terminate all workload running and to reclaim the resource quickly.

   For example, if you set the grace period to 10 seconds, workload continues to run on the borrowed resource for 10 seconds after the EGO initiates the resource reclaim. If you set the grace period to 1, any running workload is terminated almost immediately.
6. Check the Rebalance when time intervals change box.

   If you want EGO to “rebalance” or reset to the originally configured resource plan whenever a time interval change occurs (when there is a change in ownership of resources), check this box. Similarly, when resources are reclaimed (or passed back to their original owners), you can evoke a rebalancing in accordance with the original resource plan.

Before EGO rebalances according to the resource plan, a consumer's grace period is honored to help ensure workload is complete before being killed.

7. Click Apply.

Plan the ranking of your consumers knowing that resources are reclaimed based on their rank; those leaf consumers with a lower consumer rank are reclaimed before consumers with a higher rank.

CHAPTER

14

Resource Groups

Understanding resource groups

Resource groups overview

Resource groups organize a heterogeneous resource pool.

Resource groups are logical groups of hosts. Resource groups provide a simple way of organizing and grouping resources (hosts) for convenience; instead of creating policies for individual resources, you can create and apply them to an entire group. Groups can be made of resources that satisfy a specific static requirement in terms of OS, memory, swap space, CPU factor, and so on, or that are explicitly listed by name.

The cluster administrator can define multiple resource groups, assign them to consumers, and configure a distinct resource plan for each group. For example:

- Define multiple resource groups: A major benefit in defining resource groups is the flexibility to group your resources based on attributes that you specify. For example, if you run workload units or use applications that need a Linux OS with not less than 1000 MB of maximum memory, then you can create a resource group that only includes resources meeting those requirements.

  Note:

  No hosts should overlap between resource groups. Resource groups are used to plan resource distribution in your resource plan. Having overlaps causes the hosts to be double-counted (or more) in the resource plan, resulting in recurring under-allocation of some consumers.
- Configure a resource plan based on individual resource groups: Tailoring the resource plan for each resource group requires you to complete several steps. These include adding the resource group to each desired top-level consumer (thereby making the resource group available for other sub-consumers within the branch), along with configuring ownership, enabling lending/borrowing, specifying share limits and share ratio, and assigning a consumer rank within the resource plan.

Resource groups are either specified by host name or by resource requirement using the select string.

Tip:

You can use the command `egoconfig addresourceattr` to add a custom tag to any hosts and then specify that tag when creating a resource group. See the reference for more information.

By default, EGO comes configured with three resource groups: InternalResourceGroup, ManagementHosts, and ComputeHosts.

InternalResourceGroup and ManagementHosts should be left untouched, but ComputeHosts can be kept, modified, or deleted as required.

Note:

Requirements for resource-aware allocation policies (where only certain resources that meet specified requirements are allocated to a consumer) can be met by grouping resources with common features and configuring them as special resource groups with their own resource plans.

Notes on setting CPU slots

When you create a resource group in the Platform Management Console, you must choose whether to set the CPU slots in your resource group to 1 slot per CPU or *x* slots per host.

- Choose 1 slot per CPU if the workload is expected to contain long-running, compute-intensive workload units.

  Dedicating a CPU to this type of workload unit makes best use of each CPU's resources and contributes to better performance.
- Choose *x* slots per host if the workload is expected to contain short-running, I/O intensive workload units, or if workload units are not expected to be compute intensive.

  For example, choose this setting if you want more slots than CPUs for workload units that require less than one CPU to run. Setting *x* slots per host can give multiple slots per CPU when *x* is greater than the number of CPUs on the host. If *x* is less than the number of CPUs on the host (for example, setting 1 slot per host when you have many CPUs on the host), performance suffers because only one CPU gets used while other resources are wasted.

Note the following:

- There is a 1-to-1 mapping between small workload units (for example, a job, a task, a VM, etc.) and slots.
- If there is a differing individual host value of “slots per host,” it overrides the setting of *x* slots per host you set for the resource group. The host level setting overrides the group level configuration.

  For example, if there are 10 hosts in a resource group and you choose in the Console to set up 5 slots per host, you would normally expect to see 50 slots listed within the Member Host Summary section of a resource group’s properties page. However, if you see a different number showing in the summary (for example, 45), then an administrator has manually overridden the settings for one or more hosts. This individual value overrides the group setting configured in the Console.

  In some cases, even if an administrator has not manually changed the "slots per host" value, you may still see an unexpected number in the Member Hosts Summary section. This may mean that certain hosts within this particular resource group are double-allocated, meaning they are allocated to more than one resource group. In cases of double-allocation, the sum of the allocated slots displays in the Member Hosts Summary section, not the number of slots for this resource group alone. It is advised not to double-allocate slots.
- If you want to change the value of the number of slots per CPU, it must be specified on the workload management side (outside of EGO).
- The value for the number of CPUs per host is automatically detected during installation.

Lost-and-found resource groups

When host slots are allocated to a client, the vemkd detects the resource group to which the host belongs. But when the vemkd restarts, there is a brief interval (while host information gets updated) where it may not immediately detect the host’s resource group. It is during this update interval that Platform EGO creates a temporary resource group called “LOST_AND_FOUND”. The vemkd adds any host with a current allocation to this resource group if it cannot immediately detect an assigned group. Once vemkd completes its update of host information and detects the host's assigned resource group, the host automatically rejoins it.

Note:

This only happens if the host is already allocated and vemkd must trace its resource group. If the host does not currently belong to an allocation, then vemkd does not perform a search for a resource group.

Similarly, if a host with allocated slots is permanently removed from its resource group (thus never rejoining its original resource group when vemkd restarts), the vemkd adds this host to the "LOST_AND_FOUND" group. It will remain in this group until the cluster administrator frees up the allocation on the host.

Resource group properties

`ResourceGroups.xml` is located in either `$EGO_CONFDIR` (Linux) or `%EGO_CONFDIR%` (Windows). Use an XML editor to make changes to the file. The file must be valid and well formed.

Parameter name	**Description**	**Additional Information**
ResourceGroupName	The name of the resource group you specify.	Required. ManagementHosts and InternalResourceGroup are required resource groups. You can specify as many resource groups as you want. No spaces or unusual characters are allowed.
availableSlots	The number of slots you have to distribute in your resource plan for hosts in this resource group.	Optional. We recommend that you do not modify this parameter. Use to define the number of available slots for each host in a resource group selected dynamically, or for the host in the ResourceEntry when the group membership is specified with ResourceList. This value is inherited into the resource plan where you distribute the number of available slots for this resource group to your consumers. When undefined, the number of slots equals the number of CPUs on a host.
ResourceRequirement	The resources hosts require to belong to this host group.	Optional. Allows you to select only those hosts with specific static requirements to belong to this resource group. For example, you can specify only hosts with more than 2 GB of memory. You must either specify hosts by ResourceRequirement or by ResourceList. Note that the entered expression gets evaluated against each host in the cluster. If a host is found to satisfy the stated resource requirement (if it returns true/ non-zero), then the host is added to the host group. Ensure that you enter an expression related to the host and that can be used during evaluation (for example, memory requirement, swap space, temporary disk space, CPU utilization, etc.). See the section -R "res_req" in your *EGO Reference* guide for details.
ResourceList	A list of host names to belong to this host group.	Optional. Do not use with ResourceRequirement. List the host names with <ResourceEntry ResourceID=*host_name*> to belong to this host group. Host names should be separated by a space. You must either specify hosts by ResourceRequirement or by ResourceList.
ResourceEntry	Used to specify hosts by name.	Use with ResourceList to specify host names.
ResourceID	The name of the resource (for example, host name).	An attribute of ResourceEntry used to specify specific host names to belong to a resource group.

Static resources

Static resources are built-in resources that represent host information that does not change over time, such as the maximum memory 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.

The resources ncpus, maxmem, maxswp, and maxtmp are not static on UNIX hosts that support dynamic hardware reconfiguration.

Note:

You can use the command `egoconfig addresourceattr` to add a custom tag to any hosts and then specify that tag when creating a resource group. See the reference for more information.

Index	Measures	Units	Determined by
scvgf	scavenging flag	string	configuration
type	host type	string	configuration
model	host model	string	configuration
hname	host name	string	configuration
cpuf	CPU factor	relative	configuration
ncpus	number of processors	processors	lim
ndisks	number of local disks	disks	lim
maxmem	maximum memory	MB	lim
maxswp	maximum swap space	MB	lim
maxtmp	maximum space in /tmp (Linux) or OS default temp directory (Windows)	MB	lim

Scavenging flag (scvgf)

Scavenging flag is a configurable external attribute assigned to a host, identifying that it is available for scavenging. Can be turned on or off.

Host type (type)

Host type is a combination of operating system and CPU architecture. All computers that run the same operating system on the same computer architecture are of the same type. You can add custom host types in the HostType section of `ego.shared`. This alphanumeric value can be up to 29 characters long.

An example of host type is LINUX86.

Host model (model)

Host model is the combination of host type and CPU speed (CPU factor) of your machine. All hosts of the same relative type and speed are assigned the same host model. You can add custom host models in the HostModel section of `ego.shared`. This alphanumeric value can be up to 29 characters long.

An example of host model is Intel_IA64.

Host name (hname)

Host name specifies the name with which the host identifies itself.

CPU factor (cpuf)

CPU factor (frequently shortened to cpuf) is a value representing the speed of the host's CPU relative to other hosts in the cluster. For example, if one processor is twice the speed of another, its CPU factor should be twice as large. For multiprocessor hosts, the CPU factor is the speed of a single processor.

The CPU factors are detected automatically or defined by the administrator.

Number of CPUs (ncpus)

By default, the number of CPUs represents the number of physical processors a machine has. As most CPUs consist of multiple cores, threads, and processors, ncpus can be defined by the cluster administrator (either globally or per-host) to consider one of the following:

- processors
- processors and cores
- processors, cores, and threads

Globally, this definition is controlled by the parameter EGO_DEFINE_NCPUS in `ego.conf` (shared directory). The default behavior for ncpus is to consider only the number of physical processors (EGO_DEFINE_NCPUS=**procs**).

Note:

On a machine running AIX, ncpus detection is different. Under AIX, the number of detected physical processors is always 1, whereas the number of detected cores is the number of cores across all physical processors. Thread detection is the same as other operating systems (the number of threads per core).

Number of disks (ndisks)

The number of disks specifies the number of disks a machine has.

Maximum memory (maxmem)

Maximum memory is the total available memory of a machine, measured in megabytes (MB).

Maximum swap (maxswp)

Maximum swap is the total available swap space a machine has, measured in megabytes (MB).

Maximum temporary space (maxtmp)

Maximum temporary space is the total temporary space a machine has, measured in megabytes (MB).

About the selection string

The selection string is used to specify resource requirements when creating or modifying resource groups. Resource requirements are properties of a host.

The selection string can be augmented using order so you can specify which selection resources are the most important.

select

select synopsis

select(*expression***)**

select(*expressionoperatorexpression***)**

select((*expressionoperatorexpression***)***operatorexpression***)**

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_nameoperatorvalue

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, 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

Index	Measures	Units	Determined by
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	Negative of a
!	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
+	n/a	a+b	Add a and b
-	n/a	a-b	Subtract b from a
>	&gt;	a > b	1 if a is greater than b, 0 otherwise

Operator	XML Equivalent	Syntax	Meaning
<	&lt;	a < b	1 if a is less than b, 0 otherwise
>=	&gt;=	a >= b	1 if a is greater than or equal to b, 0 otherwise
<=	&lt;=	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
&&	&amp;&amp;	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)
```

Create a resource group by host names

You must be logged in as a cluster administrator and you should have already added most of your hosts to the cluster.

Create new resource groups from the Platform Management Console to ensure your consumers have the appropriate group of compute hosts available to them. Resource groups are often the easiest way to create a homogeneous group of hosts for a consumer (for example, all Linux machines). You can create a resource group by resource requirement (dynamic) or by host names (static). This procedure creates a resource group by host names.

Remember:

When you create a resource group by host names, you select specific member hosts. If any new hosts are added to the cluster, they need to be manually added to a resource group.

1. In the Platform Management Console, click Resources > Configure Resource Groups.

   Any existing resource groups are listed.
2. Select Global Actions > Create a Resource Group.
3. Specify a resource group name.

   Resource group names must consist of letters and numbers only (no spaces or special characters) and must be 64 characters or less.
4. (Optional) Include a description of the resource group.

   There is a 200-character maximum.
5. (Advanced) Specify how many slots per host you would like to have the system count.

   We do not recommend overriding the slots per host configured at the host level. It can lead to double counting your resources in your resource plan. Unless you are an advanced user, set this to 1 slot per CPU.

   The maximum number of slots per host is 9999.
6. For the Resource Selection Method, select Static (List of Names).

   Static resource selection means that you are selecting specific hosts to belong to this resource group.
7. Under Filter display of member hosts > Hosts to Show in List, select how you would like to filter your host list.
   - All hosts gives you a list of all hosts that belong to your cluster. You cannot specify any resource requirements.
   - Hosts filtered by resource requirement lets you filter your hosts and display a list of candidates for your resource group based on a set of resource requirements. For example, you can specify all hosts that are Linux.
8. If you chose to filter hosts by resource requirement, specify the resource requirement you want.

   For example, **select(LINUX86)**.

   Note the following:

- The entered expression gets evaluated against each host in the cluster. If a host is found to satisfy the stated resource requirement (if it returns true/ non-zero), then the host is added to the host group.
- Use the syntax from the selection string to specify your resource requirements. You do not need to use XML equivalents in the Platform Management Console.

Resource Requirement	Description	Example
maxmem	The maximum RAM available	select(maxmem>400)
maxswp	The maximum swap space	select(maxswp>600)
maxtmp	The maximum temporary space	select(maxtmp>100)
ncpus	The number of CPUs	select(ncpus==1)
type	The type of host	select(LINUX86)
ndisks	Number of disks	select(ndisks>1)

- Ensure that you enter a resource requirement expression that relates to the host and that can be used during evaluation (for example, memory requirement, swap space, temporary disk space, etc.).
- If you specify a Windows host name, it must be the full name not the short name.

Tip:

You can use the command `egoconfig addresourceattr` to add a custom tag to any hosts and then specify that tag when creating a resource group. See the reference for more information.

9. If you have specified a resource requirement or modified one, click Refresh Host List to get an accurate list of hosts below.
10. Expand the Member hosts section if necessary and review the hosts found.

    If you selected to filter hosts by All hosts, the list of hosts provided is all the hosts in your cluster. If you selected Hosts filtered by resource requirement, a list of hosts that currently fulfill the requirements you specified in the resource requirement string section display.
11. Review your member hosts and select the hosts you want using the checkboxes.

    If your host list is long, it may go on for several pages. You can select hosts and click Create at any time and then add more hosts from other pages. Make sure you save before navigating to another page.

    Once you have selected a member host, you can filter the list again with a different resource requirement. The hosts highlighted and check marked are your member hosts.

    By default, if you select no member hosts, all hosts in your cluster are added to this resource group when you create it. Furthermore, if you do not select a host, the resource group type switches from Static to Dynamic.
12. Click Check for overlaps.

    No hosts should overlap between resource groups. Resource groups are used to plan resource distribution in your resource plan. Having overlaps causes the hosts to be double-counted (or more) in the resource plan, resulting in recurring under-allocation of some consumers.

If any hosts overlap, remove them from this resource group or remove them from the overlapping resource group. The exception is with hosts listed in InternalResourceGroup —although all hosts in the cluster are listed here they are not “double-counted” in the resource plan.

13. Click Create.

To get back to your list of resource groups, click Resource Groups on the top of your page or Cancel at the bottom. You now need to update your resource plan based on this new resource group. Note that the number of slots available for your new resource group in the resource plan is automatically detected from what you specified in your resource group.

Create a resource group by resource requirement

You must be logged in as a cluster administrator and you should have already added most of your hosts to the cluster.

Create new resource groups from the Platform Management Console to ensure your consumers have the appropriate group of compute hosts available to them. Often resource groups are the easiest way of creating a homogeneous group of hosts for a consumer (for example, all Linux machines). You can create a resource group by resource requirement (dynamic) or by host name (static). This procedure creates a resource group by resource requirement.

Remember:

When you create a resource group based on resource requirement, you do not select specific member hosts. If new hosts that meet those requirements are later added to the cluster, they are automatically added to the resource group.

1. In the Platform Management Console, click Resources > Configure Resource Groups.

   Any existing resource groups are listed.
2. Select Global Actions > Create a Resource Group.
3. Specify a resource group name.

   Resource group names must consist of letters and numbers only (no spaces or special characters) and must be 64 characters or less.
4. (Optional) Include a description of the resource group.

   There is a 200 character maximum.
5. (Advanced) Specify how many slots per host you would like to have the system count.

   We do not recommend overriding the slots per host configured at the host level. It can lead to double counting your resources in your resource plan. Unless you are an advanced user, set this to 1 slot per CPU.

   The maximum number of slots per host is 9999.
6. For the Resource Selection Method, leave the default value Dynamic (Requirements).

   Dynamic means that the member hosts of this resource group vary according to which hosts meet that requirement. The requirement itself is static. Some commonly used requirements to create a dynamic resource group include operating system, the number of CPUs, and maximum memory.

   When you create a dynamic resource group, the new hosts added to the cluster that meet the static resource requirement you specify for the resource group are automatically added to the resource group.

   For example, you can specify a dynamic resource group with the (static) requirement of `select(maxmem>400)` and any existing hosts in the cluster that meet this requirement when you create the resource group become members and any new hosts added to the cluster afterwards that meet this requirement are members.

   Note:

   Dynamic resource requirements are not supported. A dynamic resource requirement is one that fluctuates depending on the load on the host. A dynamic resource group is created with static (unchanging) resource requirements.

7. Under Filter display of member hosts > Hosts to Show in List, select Hosts filtered by resource requirement.

   Do not select All hosts. You need to specify a resource requirement string.

   You specify a resource requirement that members of this group must meet. You can then preview those members in the next step.

8. Specify the resource requirement you want.

   For example, **select(LINUX86)**.

   Note the following:

   - The entered expression gets evaluated against each host in the cluster. If a host is found to satisfy the stated resource requirement (if it returns true/ non-zero), then the host is added to the host group.
   - Use the syntax from the select string to specify your resource requirements. You do not need to use XML equivalents in the Platform Management Console.

Resource Requirement	Description	Example
maxmem	The maximum RAM available	select(maxmem>400)
maxswp	The maximum swap space	select(maxswp>600)
maxtmp	The maximum temporary space	select(maxtmp>100)
ncpus	The number of CPUs	select(ncpus==1)
type	The type of host	select(LINUX86)
ndisks	Number of disks	select(ndisks>1)

   - Ensure that you enter a resource requirement expression that relates to the host and that can be used during evaluation (for example, maximum memory requirement or maximum swap space).
   - If you specify a Windows host name, it must be the full name not the short name.

   Tip:

   You can use the command `egoconfig addresourceattr` to add a custom tag to any hosts and then specify that tag when creating a resource group. See the reference for more information.

   By default, if you specify no resource requirement, all hosts in your cluster are added to this resource group when you create it.

9. Click Refresh Host List and expand the Member hosts section if necessary.

   A preview list of hosts that currently fulfill the requirements you specified in the resource requirement string section displays. This member list is dynamic and reflects the hosts that are currently on your cluster that meet this criteria. If hosts are added or deleted that meet the requirement you specified, they are automatically added or removed from this list.

10. Review your member hosts.

If you do not like the list of hosts found, refine your resource requirement selection string and try again.

11. Click Check for overlaps.

    No hosts should overlap between resource groups. Resource groups are used to plan resource distribution in your resource plan. Having overlaps causes the hosts to be double-counted (or more) in the resource plan, resulting in recurring under-allocation of some consumers.

    If any hosts overlap, remove them from this resource group or remove them from the overlapping resource group. The exception is with hosts listed in InternalResourceGroup—although all hosts in the cluster are listed here they are not “double-counted” in the resource plan.

12. Click Create.

To get back to your list of resource groups, click Resource Groups on the top of your page or Cancel at the bottom. You now need to update your resource plan based on these new resource groups. Note that the number of slots available for your new resource group in the resource plan is automatically detected from what you specified in your resource group.

View resource groups and their properties

Your cluster must be up and running.

View default and custom resource groups in the Platform Management Console to see their properties.

1. In the Platform Management Console, click Resources > Configure Resource Groups.

   Any existing resource groups are listed.
2. Locate the resource group you want properties for.

   Some resource group information is provided inline with the resource group name. For example, you can see the type of resource group (static or dynamic), a description (if you have provided one), the number of member hosts, and the number of slots you selected.
3. To see more details, click on the resource group name you would like to view.

   The resource group properties display. You can also modify your resource group from here.
4. Review your settings for your resource group.

   Make sure the Member hosts section is expanded so you can view the current member hosts. If you selected a static resource group, your member hosts are highlighted in yellow and have checkmarks. If you selected dynamic resource groups, the hosts listed here are members as of the last refresh point.

View hosts in a resource group

You must be logged in as a cluster administrator.

If you need to know what hosts belong to which resource group, you can view the hosts in your resource group using the Platform Management Console.

1. Click Resources > Configure Resource Groups.

   A list of existing resource groups displays.
2. Click the name of the resource group you want.

   The resource group properties display.
3. Look at the Member hosts section.

   Static hosts that belong to this resource group are highlighted in yellow and have modification check boxes beside them. You can modify static member hosts at this time if you like.
4. Click Check for overlaps to see if any of the hosts belong to other resource groups.

   Your hosts should never belong to more than one resource group. Modify your resource groups until there are no overlaps. The exception is with hosts listed in InternalResourceGroup—although all hosts in the cluster are listed here they are not “double-counted” in the resource plan.

Remove a resource group

You must be logged in as a cluster administrator.

You cannot remove a resource group if it is being used in the resource plan. Modify your plan until no resources are allocated from the resource group you want to remove.

Remove a resource group if it no longer meets your needs or is not being used. Hosts should not belong to multiple groups, so when you create new groups, you should modify or delete the old groups.

1. In the Platform Management Console, click Resources > Configure Resource Groups.

   Any existing resource groups are listed.
2. Locate the resource group you want to remove and select Actions > Delete Resource Group.
3. Confirm that you want to delete the resource group.

Using the Platform Management Console, check that your resource plan is correct for all remaining resource groups.

Modify resource groups

You must be logged in as a cluster administrator.

Modify resource groups so that you can refine your resource allocations used in your resource plan. You may also want to add or remove hosts from resource groups if you have recently added or removed hosts in your cluster.

1. In the Platform Management Console, click Resources > Configure Resource Groups.

   Any existing resource groups are listed.
2. Click the name of the resource group you want to modify.

   The resource group settings display.
3. Change any settings you want.

   If you select Static (List of Names), follow the guidelines in the topic Create a resource group by host names.

   You cannot modify "number of slots per host" for a group with static allocation while any of the group’s resources are allocated to consumers.

   If you select Dynamic (Requirements), follow the guidelines in the topic Create a resource group by resource requirement.

   You cannot modify a group with dynamic membership while any of the group’s resources are allocated to consumers.
4. When you are satisfied with your settings, click Apply.

Using the Platform Management Console, check that your resource plan is correct considering any changes to the resource groups you just made.

CHAPTER

15

The Resource Plan

Resource plan: an overview

The resource plan defines how cluster resources are allocated among consumers. The plan takes into account the differences between consumers and their needs, resource properties, and various other policies concerning consumer rank and the allocation of resources.

In general, the cluster administrator uses the SDK to first define consumer conditions that trigger a request to EGO for additional resources (for example, if an application response time or a queue depth exceeds some predetermined threshold). When a request from a consumer comes in for additional resources, EGO uses the resource plan to determine whether or not to allocate resources to that consumer. The resource plan also controls the quantity of resources that EGO allocates to the consumer at any time.

A resource plan supports a number of rules used to set policies for how, when, and where resources get allocated. This includes policies concerning resource allocation and ownership, borrowing and lending, sharing, and the reclaiming of borrowed resources. In all cases, resource plans must be set at the leaf level (consumers with no descendents).

Allocation types

EGO allocates resources to a consumer through ownership, borrowing, or sharing. A consumer can use one or a combination of these methods to secure resources. All are a form of allocation.

- Ownership refers to the guaranteed allocation of a minimum number of resources to a consumer.
- Borrowing refers to the temporary allocation of owned resources from a lending consumer to a consumer with an unsatisfied demand.
- Sharing refers to the temporary allocation of unowned resources from a “share pool” to a consumer with an unsatisfied demand.

Dynamic allocation overview

For dynamically allocated resources, client demand is fundamental to the allocation. If there is competition from other consumers, or limits configured for the consumer, the allocation could be less than what is needed. This is considered a shortfall.

The priority for allocation is to satisfy each consumer's reserved ownership, then allocate remaining resources to consumers that have demand. By default, resources are systematically allocated in the following order:

1. EGO always allocates owned resources first to consumers according to the resource plan.
2. Then, when there are no owned resources left to borrow from consumers who are willing to lend them, EGO allocates unowned resources from the share pool to consumers with unsatisfied demand.

    - Share pool resources first come from the “family” pool (any unowned resources within a particular branch in the consumer tree).
    - Once the family pool is exhausted, then EGO distributes resources from other branches in the consumer tree, eventually moving up the tree to distribute any unowned resources from the cluster level.
    - Share pool resources are distributed to competing leaf consumers according to their allowed share ratio. If the share ratio for two competing consumers is 1:1, resources are allocated evenly. If the share ratio is 1:2, then the second competing consumer gets twice as many available resources from the share pool.

3. Finally, when there is demand, consumers lend and borrow resources from each other.

Note:

If a consumer has previously lent out resources to another consumer, the lending consumer recalls the owned resources necessary to meet its current unsatisfied demands once other allocation options have been exhausted, regardless of the borrowing consumer’s priority. This default behavior can be changed so that owned resources are instead recalled first before borrowing from other consumers.

Resource update cycle

By default, EGO updates the resource information used for scheduling every 60 seconds. The frequency of the update cycle is determined by EGO_RESOURCE_UPDATE_INTERVAL in `ego.conf`.

The update cycle determines how quickly EGO detects a new resource or unavailable resource in the cluster. It also determines how often EGO detects changes in load indices that might be used to allocate resources to jobs.

Customizing the resource plan

A general order of events for customizing a resource plan might be as follows:

1. If resource groups are necessary, create them first.
2. Create consumers and edit the plan.
3. Customize the resource plan for each consumer.
4. Create a resource plan for each new resource group you add.

The plan can be modified by the cluster administrator and the consumer administrator. The consumer administrator has restricted permissions.

Out-of-box resource plan

EGO installs with two top-level consumers: ManagementServices and SampleApplications. Do not remove the ManagementServices branch: it contains required system services.

The default resource plan can be modified as required (except for removing the ManagementServices branch). You need to create a consumer for each new application.

The provided sample consumer, EclipseSamples, allows you to begin using EGO right away. This consumer comes with registered applications that are ready to run. You must have the EGO SDK installed.

Important:

If you want to keep the default plan intact (to reset back to it at a later time), be sure to export and save it locally before making any changes. Import it when required.

Reserving management host slot to run services

By default, there are 12 slots per management host. Although this number is configurable, be aware that each service requires at least 1 slot to run. For example, if you configure the EGO Services leaf consumer with 0 owned slots, the cluster does not run. Note that slot ownership must filter down through the consumer tree until a service inherits a slot directly from a parent. To avoid service interruption and to protect a service’s owned resource, ensure the following:

- Disable borrowing and lending on the service slot.
- Set the share ratio to 1.
- Set the priority uniformly across all consumer parent/child levels for each service.

Owning and borrowing resources

Ownership

When consumers “own” resources, they are guaranteed of a minimum allocation of resources, regardless of competition from other consumers. Ownership is expressed as a numeric quantity.

Ownership is optional. A consumer may not own any resources yet still use cluster resources allocated to them through borrowing. Consumers can choose to lend idle resources.

Overallocating at the cluster level

It is possible to allocate to the cluster(at the top level only in your resource plan) more slots than are currently listed as available. Only do so if resources are, for some reason, not available when you modify your resource plan, but would usually be part of the cluster and will be available when the time interval you are overallocating for occurs.

Attention:

Only overallocate ownership at the cluster level if you know that resources will be available during the time interval you are setting it for.

Lending resources

Lending is optional. You can enable lending only for leaf consumers who own resources (there are no lend settings available for non-leaf consumers in the resource plan). During periods of low demand, a consumer's resources can be lent to other consumers who have an unsatisfied demand. This kind of resource lending/borrowing relationship between consumers improves the efficiency of the cluster. Without lending, owned resources cannot be shared with other consumers and idle resources are wasted.

Owned resources that are not being used and that have lending enabled get allocated to consumers who have an unsatisfied demand. Qualifying resources are lent in the order of configured consumer rank. For example, in the case where a consumer has resources available to lend, and there are competing consumers with unsatisfied demand, this is what would happen:

- First, the borrowing consumer with the highest assigned consumer rank is allocated as many resources as are available until its demand is satisfied or until its configured borrowing limit is reached.
- Then, any surplus resources are assigned to the competing consumer with the next highest consumer rank.
- The allocation continues down the line of consumer rank until all qualifying resources are allocated or all consumer demands are satisfied.

Lending can occur between consumer branches in the consumer tree, and is not restricted to leaf consumers from the same consumer branch. However, through advanced refinement of the resource plan, leaf consumers can be configured to only lend to and borrow from their siblings.

EGO reclaims resources from a borrowing consumer and returns them to the lending consumer as soon as the lending consumer has an unsatisfied demand. Although ownership of resources guarantees access to them at any time, preconfigured reclaim grace periods may delay the recovery of lent resources. When a cluster or consumer administrator sets the reclaim grace period for a consumer, they should consider the length of a typical workload unit potentially run by a borrowing consumer, along with the urgency of workload units that need to be done by a lending consumer who must reclaim its resources.

Reserving resources by setting a lending limit

A consumer has the flexibility to enable lending on all of its owned resources, or on only a few; those resources without lending enabled are reserved solely for use by the leaf consumer that owns them. The reserved resources do not qualify for lending and are never lent out, even if unused. The lending limit is expressed as a numeric quantity.

Borrowing and sharing

Borrowing refers to the temporary allocation of owned resources from a lending consumer or the share pool to a consumer with an unsatisfied demand.

Sharing refers to the temporary allocation of unowned resources from a “share pool” to a consumer with an unsatisfied demand.

Any client can make use of unused, owned resources that are enabled for lending. The only unused resources that cannot be borrowed are those that are reserved for use solely by a resource owner (that is, resources belonging to a consumer who has not enabled lending).

Borrowing is optional. If borrowing is disabled, the allocation to a leaf consumer never exceeds the configured ownership. Therefore, if borrowing is disabled for all consumers, any unused resources (owned by other consumers) are wasted.

Borrowing resources is on a first-come first-served basis. For example, one leaf consumer can borrow all the available resources in the cluster by being the first to request them. Once all available resources are allocated, other leaf consumers that want to borrow must then wait for a resource to be released.

Borrowing can be enabled for leaf consumers only.

Sharing resources between leaf consumers from the same consumer branch

In cases where leaf consumers from the same consumer branch are competing to borrow resources from the share pool, the share ratio determines the minimum number of resources to allocate to each of them.

The share ratio is configurable. A valid entry for a share ratio is a positive, whole number. Share ratios work in this way:

- By default, all consumers have a share ratio of 1, meaning they share equally.
- A share ratio of 0 (zero) means that a consumer cannot borrow at all from the share pool.
- A leaf consumer with a ratio of 2 can borrow twice as many resources as a competing sibling with a ratio of 1, and half as much as a competing sibling with a ratio of 4.

Other examples of share ratios between competing leaf consumers (siblings):

- Scenario: Two competing leaf consumers (siblings) with equal allocations

A ratio of 1:1 means that both siblings receive 1/2 of the available resources from the parent.

- Scenario: Two competing leaf consumers (siblings) with unequal allocations

A ratio of 1:2 means that one sibling receives 1/3 of the available resources from the parent while the other sibling receives 2/3 of it.

- Scenario: Ten competing leaf consumers (siblings) with equal allocations

A ratio of 1 each means each sibling receives equal resources (1/10th of the parent's available resources).

Note:

Resource allocation to competing leaf consumers depends in turn on branch ownership and share ratios. Therefore, for consumers on different branches of the consumer tree, an identical share ratio does not imply an identical allocation of resources.

In addition to setting share ratios, the cluster administrator may set maximum shares for each consumer. A maximum share value is specified as an absolute numerical count of resources.

Share ratio enforcement throughout the consumer tree

By default, planned share ratios are enforced at the leaf level. This means that share policies guarantee that each application (registered at a leaf level) receives its planned or "deserved" number of resources when demand is demonstrated. If an application does not have sufficient demand to warrant receiving all its deserved resources, the unused resources are distributed to all consumer branches and filtered down to leaf consumers as per their relative share ratios.

You can change the default behavior to enforce share ratios at the parent level. Doing this forces EGO to distribute unused resources to sibling leaf consumers (within a single line of business) that exhibit demand first before it distributes them throughout the rest of the consumer tree (to other lines of business). This allows a line of business to share resources between its own registered applications before sharing with other lines of business.

Resource allocation according to consumer rank and borrowing preference

Once a consumer branch's share pool of resources is exhausted, then EGO allocates resources from other branches in the consumer tree, eventually moving up the tree to allocate any unowned resources from the cluster level.

Leaf consumers borrow resources from other consumer branches according to the following policies:

1. Consumer ranking: Leaf consumers from the same consumer branch with the highest priority setting have the first opportunity to borrow.

   Note:

   The cluster administrator can set a maximum number of resources that can be borrowed by each consumer.

2. Borrowing preference order: In cases where resources may be borrowed from multiple sources, lenders are ordered by "borrowing preference". A borrower's demands are first satisfied by borrowing from the lender for which he has the highest borrowing preference.

Limited borrowing

By default, a consumer with unsatisfied demand can potentially borrow all qualifying resources. However, you can choose to limit the number of borrowed resources allocated to a specific consumer. The borrowing limit is expressed as a numeric quantity.

In cases where a consumer owns resources and also borrows additional resources, the specified maximum allocation includes both the borrowed and owned resources.

Resource reclaim

A consumer does not retain guaranteed use of borrowed resources. Borrowed resources get returned to their owners in two situations:

- When the borrowing consumer and its client releases them
- When owners reclaim their resources to meet their own unsatisfied demand

Resource reclaim is influenced by the grace period set by cluster or consumer administrators and the configured consumers rank.

Note:

EGO may not always return the exact resource that was originally lent. In cases where a high priority workload unit may be running on a lent resource, an analogous resource may be returned instead to the original lending consumer. This behavior is dependent upon the application manager or consumer (for example, Platform Symphony or an LSF cluster) that may be installed on EGO.

Grace period

Lent resources can be reclaimed by owners experiencing unsatisfied demand even if the client is using them. When a resource is reclaimed, any client workload units running on the resource are interrupted. You can set a grace period, however, to impose a delay before a borrowed resource is returned to its owner. For example:

- If you set the grace period to 10 seconds, any client workload units continue to run on the borrowed resource for 10 seconds after EGO initiates the resource reclaim.
- If you set the grace period to 1, any running client workload units are almost immediately interrupted.

Before setting a grace period, consider the length of a typical workload unit that is run by a borrowing consumer and its clients, and the urgency in which a lending consumer might require its demands be satisfied.

Note:

Leaving the grace period blank or specifying 0 uses the default grace period of 120 seconds.

Reclaim according to consumer rank

Resources are reclaimed according to their configured consumer rank.

- Example 1: If a lending consumer has unsatisfied demand and requires that its lent resources be reclaimed, EGO looks to reclaim resources starting with leaf consumers with the lowest consumer rank.
- Example 2: If a lending consumer has a specific resource requirement (for example, the lending consumer needs a Windows slot with a certain amount of available memory), EGO reclaims the first lent resource it finds that matches this requirement. Borrowing leaf consumers with the lowest consumer rank are considered first, followed by leaf consumers with a higher consumer rank.

Note:

Consumer rank is only considered when resources are reclaimed.

Change reclaim behavior for owned resources

By default, owned resources are only reclaimed after the lending consumer has attempted to satisfy its unmet demand through all other available means, including by borrowing resources from other lending consumers. You can, however, change this behavior so that owned resources get reclaimed before a consumer attempts to borrow resources from other lending consumers.

Changing the reclaim behavior is useful in cases where a consumer's owned resources are specially selected to run certain workload units, or in charge-back settings where borrowing from outside a resource group might be more costly.

Change share pool reclaim behavior

By default, share pool resources can be reclaimed. This allows the share pool to reclaim resources from an over-allocated consumer to meet the demands of a competing consumer with a higher share ratio. You can change this behavior so that share pool resources are not reclaimed. Instead, resources get returned to the share pool for further allocation once the borrowing consumer and its client releases them.

Troubleshooting unexpected resource allocation issues

If you find that leaf consumers are not getting enough resources, or that client workload units are not running as expected, check the following:

- Ensure that the entire consumer branch owns adequate resources (that parents own enough resources to meet the demands of their children).
- Check that the priority levels are set appropriately (that they are not all set to “low” or all set to “high”).
- Confirm that the share ratio is appropriate between sibling leaf consumers (that more important leaf consumers are given a higher share ratio than competing siblings).
- Make sure that you enable borrowing and lending.

Share ratio behavior and enforcement

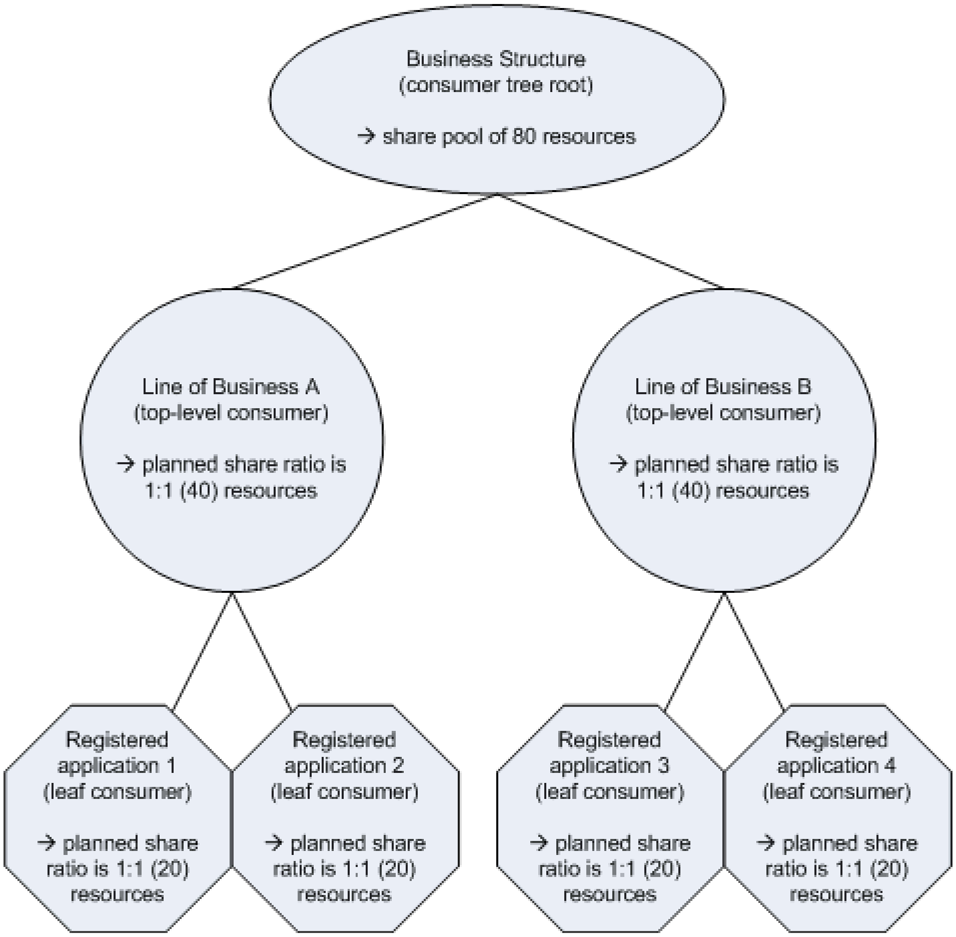
Sharing refers to the temporary allocation of unowned resources from a “share pool” to a consumer with an unsatisfied demand. Behavior of how these resources are distributed to consumers can be configured.

Default behavior—share ratio enforced at leaf level

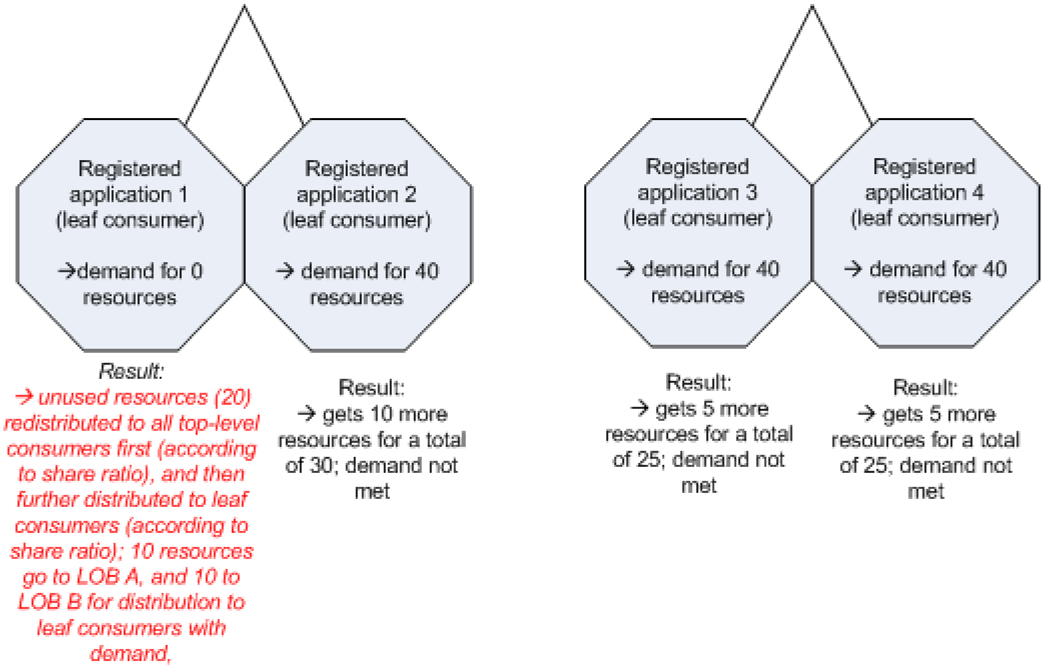
Distribution model

By default, planned share ratios are enforced at the leaf level. This means that existing share policies guarantee that each application (registered at a leaf level) receives its planned or “deserved” number of resources when demand is demonstrated. If an application does not have sufficient demand to warrant receiving all its deserved resources, the unused resources are distributed to all consumer branches and filtered down to leaf consumers as per their relative share ratios.

In the following sample resource distribution model, all applications (registered to leaf consumers) are configured with equal share ratios; each leaf consumer has a 1:1 share of the resources distributed to its parent (top-level consumer from the same branch). Assuming all applications have the same demand, all receive the same number of resources.


Business Structure
(consumer tree root)
→ share pool of 80 resources
Line of Business A
(top-level consumer)
→ planned share ratio is 1:1 (40) resources
Line of Business B
(top-level consumer)
→ planned share ratio is 1:1 (40) resources
Registered application 1
(leaf consumer)
→ planned share ratio is 1:1 (20) resources
Registered application 2
(leaf consumer)
→ planned share ratio is 1:1 (20) resources
Registered application 3
(leaf consumer)
→ planned share ratio is 1:1 (20) resources
Registered application 4
(leaf consumer)
→ planned share ratio is 1:1 (20) resources

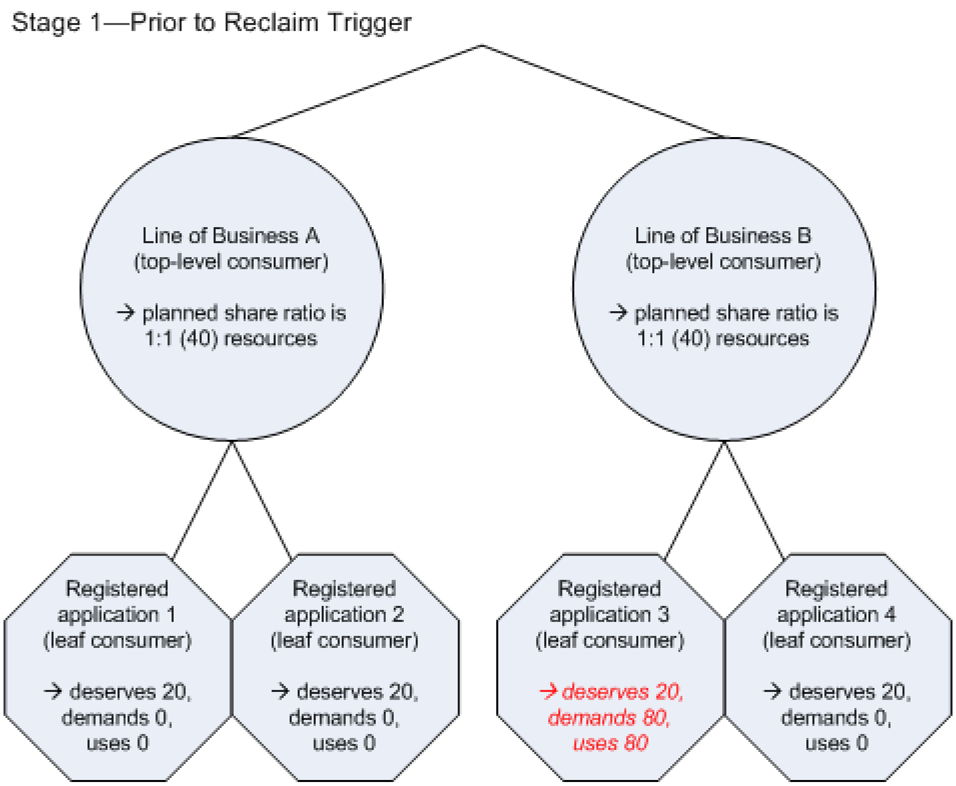
Building on the sample resource distribution model above, if one of the applications no longer demonstrates a resource demand, its “deserved” resources are distributed to other leaf consumers in the tree according to their configured share ratios (in this case, the share ratios are equal). This is the default behavior.

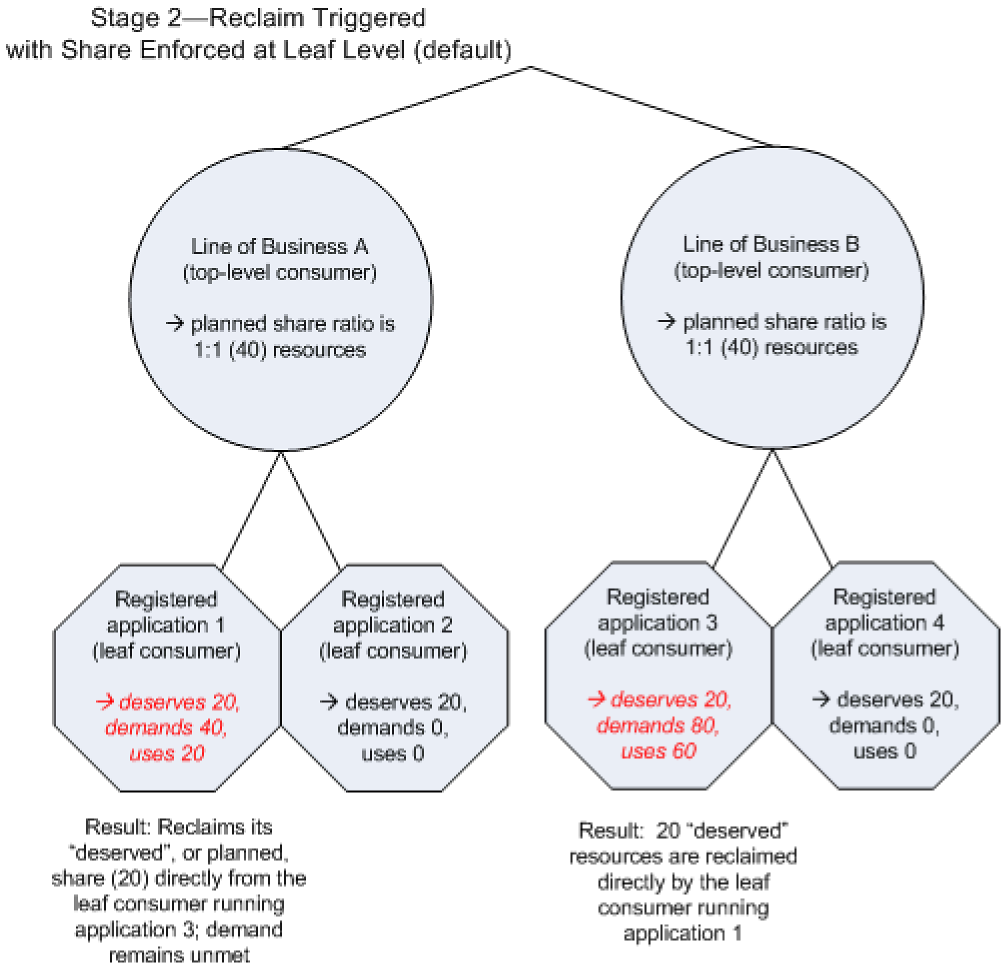

Registered application 1 (leaf consumer)
→demand for 0 resources
Registered application 2 (leaf consumer)
→ demand for 40 resources
Registered application 3 (leaf consumer)
→ demand for 40 resources
Registered application 4 (leaf consumer)
→ demand for 40 resources
Result:
→ unused resources (20) redistributed to all top-level consumers first (according to share ratio), and then further distributed to leaf consumers (according to share ratio); 10 resources go to LOB A, and 10 to LOB B for distribution to leaf consumers with demand,
Result:
→ gets 10 more resources for a total of 30; demand not met
Result:
→ gets 5 more resources for a total of 25; demand not met
Result:
→ gets 5 more resources for a total of 25; demand not met

Reclaim behavior

If reclaim is triggered, a leaf consumer takes back its “deserved” number of resources in use by other consumers, up to its planned share. For example, if a leaf resource is experiencing an unmet demand, then it reclaims resources directly from another leaf consumer who is using more than its planned share of resources.

The first leaf consumer reclaims without consideration of the needs or planned share ratio of the second leaf consumer’s branch. The first leaf consumer does not care if the consumer branch it reclaims from falls below the branch’s “deserved” number of resources.

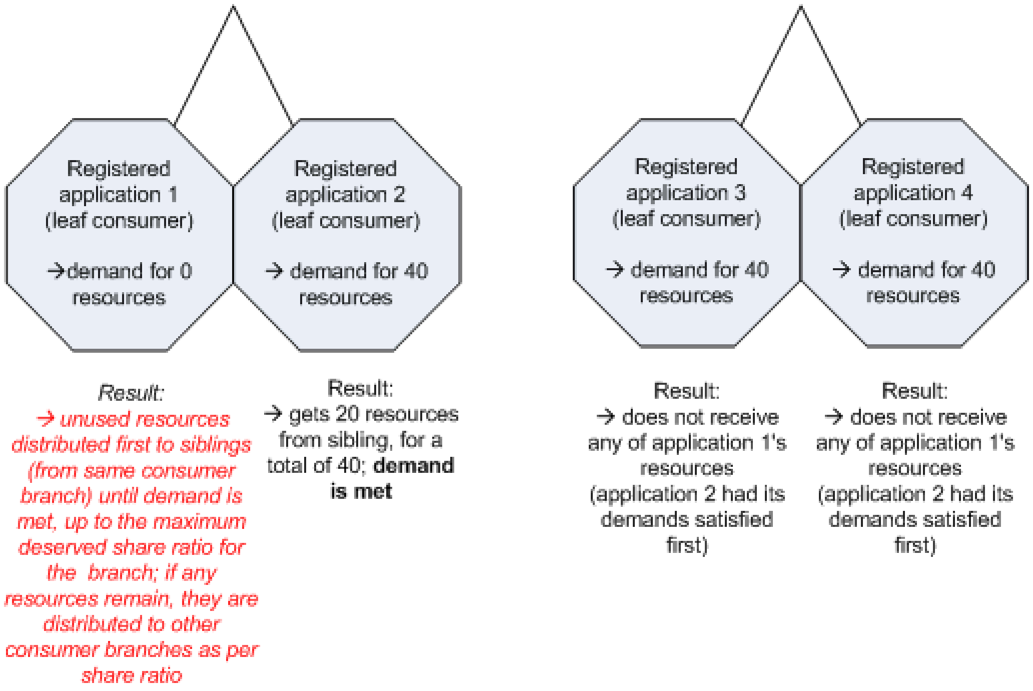

Stage 1—Prior to Reclaim Trigger
Line of Business A (top-level consumer)
→ planned share ratio is 1:1 (40) resources
Line of Business B (top-level consumer)
→ planned share ratio is 1:1 (40) resources
Registered application 1 (leaf consumer)
→ deserves 20, demands 0, uses 0
Registered application 2 (leaf consumer)
→ deserves 20, demands 0, uses 0
Registered application 3 (leaf consumer)
→ deserves 20, demands 80, uses 80
Registered application 4 (leaf consumer)
→ deserves 20, demands 0, uses 0



Reconfigured behavior—share ratio enforced at parent level

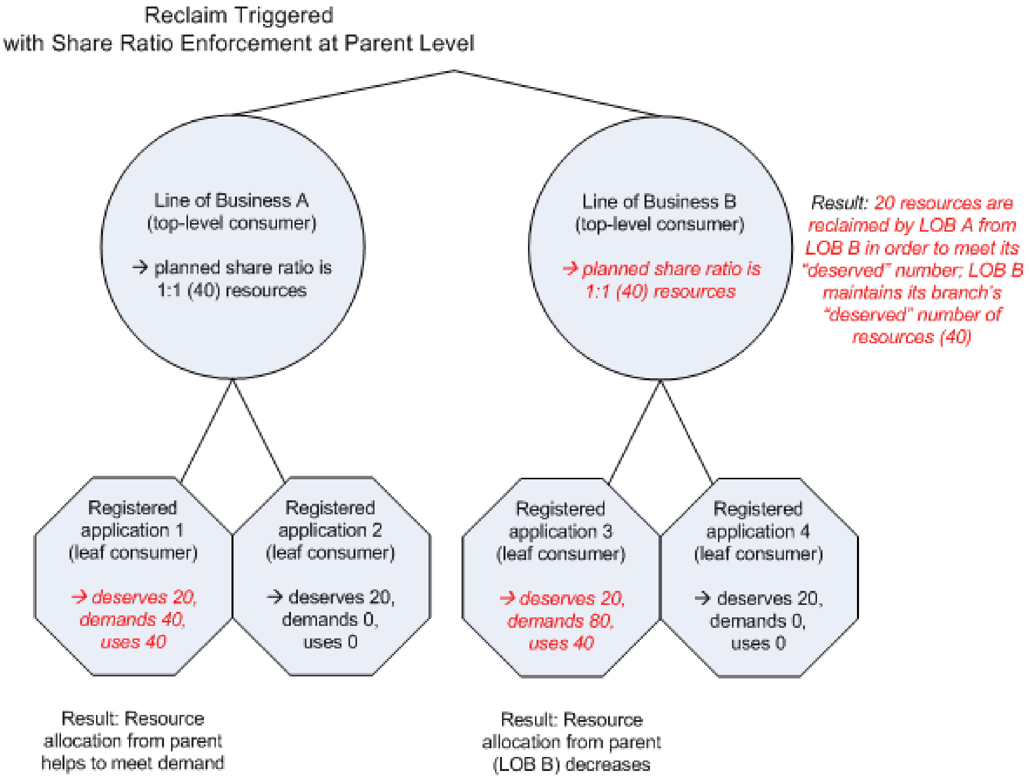
Distribution model

EGO can be configured so that consumers' share ratios are enforced at the parent level instead of at the leaf level. This means that if one of the applications no longer demonstrates a resource demand, its "deserved" resources are distributed first to its siblings (other leaf consumers with registered applications within the same consumer branch) experiencing an unmet demand, and then to other leaf consumers in the tree.


Registered application 1 (leaf consumer)
→demand for 0 resources
Registered application 2 (leaf consumer)
→ demand for 40 resources
Registered application 3 (leaf consumer)
→ demand for 40 resources
Registered application 4 (leaf consumer)
→ demand for 40 resources
Result:
→ unused resources distributed first to siblings (from same consumer branch) until demand is met, up to the maximum deserved share ratio for the branch; if any resources remain, they are distributed to other consumer branches as per share ratio
Result:
→ gets 20 resources from sibling, for a total of 40; demand is met
Result:
→ does not receive any of application 1's resources (application 2 had its demands satisfied first)
Result:
→ does not receive any of application 1's resources (application 2 had its demands satisfied first)

Reclaim behavior

By enforcing share ratios at the parent level, a consumer branch's "deserved" number of resources is considered in cases where reclaim is triggered. Resources are no longer reclaimed directly from a leaf consumer, but are instead reclaimed from the top-level consumer (parent). A parent only releases shared resources if it does not cause unmet demand in its own leaf consumers; if there is demand in its own branch, a parent does not release more resources than its "deserved" share requires.


Reclaim Triggered
with Share Ratio Enforcement at Parent Level
Line of Business A
(top-level consumer)
→ planned share ratio is 1:1 (40) resources
Line of Business B
(top-level consumer)
→ planned share ratio is 1:1 (40) resources
Result: 20 resources are reclaimed by LOB A from LOB B in order to meet its "deserved" number; LOB B maintains its branch's "deserved" number of resources (40)
Registered application 1 (leaf consumer)
→ deserves 20, demands 40, uses 40
Registered application 2 (leaf consumer)
→ deserves 20, demands 0, uses 0
Registered application 3 (leaf consumer)
→ deserves 20, demands 80, uses 40
Registered application 4 (leaf consumer)
→ deserves 20, demands 0, uses 0
Result: Resource allocation from parent helps to meet demand
Result: Resource allocation from parent (LOB B) decreases

Time-based resource planning

Time-based resource distribution

The resource plans can change according to time of day.

Time-based configuration in the resource plan allows the resource distribution for a consumer to change according to the time of day.

Time interval boundaries are at the hour point, so the minimum time interval is 60 minutes. The daily pattern repeats itself every 24 hours.

The simplest configuration is to divide the 24-hour period into 2 intervals such as daytime and nighttime, but you can specify multiple time intervals in the plan, with different resource distribution during every interval.

Example

For example, a consumer requires an ownership of 20 slots from 8:00 a.m. to midnight and only 10 slots from midnight to 8:00 a.m. Define two time intervals: one with ownership=20, one with ownership=10.

If the consumer wants the option to borrow resources only from 8:00 a.m. to 4:00 p.m., three time intervals are required:

- midnight-8:00 a.m. (ownership=10, borrowing disabled)
- 8:00 a.m.- 4:00 p.m. (ownership=20, borrowing enabled)
- 4:00 p.m.- midnight (ownership=20, borrowing disabled)

Resource distribution changes

When a new time interval begins, with one or more consumers having new values for ownership or borrowing configuration, EGO responds to the change in plan immediately.

The change in plan can trigger reclaim of resources. Examples of how plan changes can affect consumers include the following:

Increased ownership	Decreased ownership	Decreased share pool
A consumer whose ownership increases when a new time interval begins may find themselves suddenly under-allocated. If there are workload units pending, EGO distributes more resources immediately.	A consumer whose ownership decreases when a new time interval begins could have workload units interrupted as resources get redistributed or reclaimed by other consumers. When a new time interval begins, resources are immediately reallocated with changes in ownership, regardless of demand considerations. After the new time interval's ownership allocations are made, resources may be reallocated back to consumers with demand through configured borrowing or sharing policies.	In cases where the resource distribution model changes from full share (with no ownership) to a hybrid model (with both sharing and resource ownership), a decrease in share pool resources between plans or given time intervals could affect configured consumer allocations. If there are not enough unallocated resources in the share pool at the moment a new resource plan or time interval takes effect, then consumers are not allocated the planned number of owned resources. Allocated resources from the share pool must first be released back to the share pool by the clients that are using them before they can be reallocated as owned resources to a consumer.
	A consumer whose ownership decreases when a new time interval begins could continue running workload units without interruption in this situation: if borrowing is enabled, the resource status can change from owned to borrowed.	

Resource sharing: best practices

Overview: Moving from a silo to a distributed sharing model

Novice EGO administrators may implement a “silo” model for initial testing, creating resource groups that remain quite distinct and self-contained, and configuring resource plans to exclude the ability to borrow and lend resources between consumers. By maintaining such a limited model, administrators lose the advantages of EGO’s flexibility to dynamically respond to consumer and client needs and to distribute resources effectively across a cluster. To break down the silo and move to a more distributed sharing model, you must enable lending and borrowing. This can be done by altering the resource plan in a variety of ways, and ranking consumers.

Suggested best practices for resource planning include:

- Configuring resources that are available for sharing
- Configuring surplus resources for lending

Note that the key to effective resource distribution includes more than configuring suitable plans and policies. You must also ensure that there are enough resources distributed to the resource group(s) in order for the lend/borrow plans to make a difference. A transition from a silo to a distributed model of resource distribution requires a combination of effective resource plan configuration and the appropriate distribution of resources to resource groups in the tree.

Configuring resources that are available for sharing

The cluster administrator may set aside a number of resources for all consumers within the consumer tree to share. These unowned resources get distributed throughout the tree according to share entitlements (share ratio).

A best practice to help you better leverage the distribution of resources available for sharing is to interpret the share ratio as a minimum share value. These minimum values are complemented by the maximum share values you must also set. A distribution policy that includes resources available for sharing should always yield actual share values between the minimum and maximum constraints. So long as this remains the primary expectation, the complexities of the actual distribution algorithm do not matter.

Creating a resource plan for lending and borrowing

Some questions and options to consider when creating your resource plan, presented from a consumer's perspective.

Questions for leaf consumers	**Summary of options for leaf consumers based on current configuration**	
	No	**Yes**
Do you own resources?	You can still borrow resources. Be sure to enable borrowing. Be sure other consumers agree to lend to you.	A leaf consumer is allocated resources directly from its parent.
As an owner of resources, have you enabled lending?	Your resources can only be used by you. Unused resources are wasted and never lent out.	You can reclaim borrowed resources later. You can also set a lending limit so that only some unused resources get lent, and others remain available to you at all times. You can choose to lend to any consumer that wants to borrow or you can specify which consumers you want to lend to.
Do you want to restrict lending to leaf consumers from the same branch only?	When you set your lending preferences, do not restrict them to leaf consumers from the same branch (siblings).	When you set your lending preferences, list only those leaf consumers from the same branch as preferred consumers that you lend to.
Does your branch have surplus, unowned resources?	Your parent has distributed all of its resources between you and the other leaf consumers in your branch.	Surplus resources can be borrowed by you and other leaf consumers when required. Parent consumers are not able to reclaim surplus resources, but they are returned when the demand by borrowers subsides. Parents may choose to create an "imaginary" child to assign surplus resources to allow for more flexibility (to allow for reclaim and other resource planning).
Does your parent consumer have a shortage of resources?	All leaf consumers are allocated resources to match the resource plan. If demand increases, you can borrow more resources from other consumers who have unused resources available for lending.	Parents who do not have enough resources available to meet the demands of their children allocate what they do have to their highest priority child. Once its demands are met, the next highest priority child receives what is left of the parent's resources, and so on.

Questions for leaf consumers	Summary of options for leaf consumers based on current configuration	
	No	**Yes**
Do you have an unsatisfied demand?	If you have unused resources with lending enabled, they can be borrowed by other consumers experiencing demand.	If you are using all of your owned resources and you still require more, you can borrow surplus resources from other leaf consumers or from the branch-level. Once you have exhausted borrowing and sharing options, you can reclaim those resources that you have lent to other consumers*. If there is a reclaim grace period set by the borrower of your resources, you may have to wait until the grace period expires or until the borrowing consumer finishes running workload units before getting it back. **Tip:** *This default behavior can be changed so that owned resources get reclaimed before a consumer attempts to borrow resources from other lending consumers.
Are there several consumers with a simultaneous demand?	Demand is low, with no competition for resources. Consumers are using their owned resources, or borrowing unused resources without dispute.	If you and a sibling both have an unmet demand for resources, than the share ratios for each of you are considered before your parent allocates surplus resources; if you have a higher share ratio, you receive more of the surplus resources. You may have a maximum limit set for the number of resources you can receive, despite your share ratio. If there is competition between siblings for resources belonging to another consumer branch, the leaf consumer with the highest consumer rank has its demand satisfied first. In cases where the consumer rank is the same, borrowing preference are considered. For example, if the lender is configured to show a higher borrowing preference towards you, then you have your resource demands satisfied first.

Questions for leaf consumers	Summary of options for leaf consumers based on current configuration	
	No	Yes
Are you borrowing resources?	Demand may be low. Ensure you enable borrowing in case you have an unsatisfied demand in future.	Borrowing is on a first-come first-served basis. If there are competing borrowers for a consumer’s resources, and you are the first one to request them, you get as many as required to meet your demand. Any remaining resources go to the borrowers in line behind. If there are no resources left, competing borrowers must wait for you to release your borrowed resources (once your demand subsides). As a borrower, there is the possibility that any running workload units may be interrupted if the lender of your resources reclaims them. You have the option of setting a reclaim grace period to avoid interrupting any workload units running on a borrowed resource. Resources meeting the demand requirements of the lender are reclaimed according to consumer rank, from lowest to highest.
As a borrower, have you set a grace period to extend how long you can use borrowed resources before they are reclaimed?	If you are a borrower, your borrowed resource gets reclaimed as soon as the lender has demand. Any client work running on the resource is interrupted.	If you are a borrower and the lender wants their resource back to meet their own demand, any running workload units continue to run uninterrupted until the reclaim grace period expires, or until the client releases the resource (whichever comes first).
As a borrower, do you have a borrowing limit?	You can borrow as many owned resources available for sharing as you need to meet your resource demands.	The number of total resources you can use (including owned, borrowed, and shared resources) has a set limit. If your demand is great, you may not be allowed to satisfy it completely.
Are your client’s workload units running as expected? Are your resource demands consistently being met?	• Ensure that the entire consumer branch in the consumer tree is adequately resourced. • Check that priority levels are not all set the same (all “low” or all “high”). • Confirm that the share ratio between sibling leaf consumers is appropriate. • Make sure borrowing and lending are enabled.	The resource plan is effective.

Configure EGO to enforce share ratio at parent level

Share ratios are enforced at the leaf level by default. To change the configuration so that share ratios are enforced at the parent level, you need to set a parameter in `ego.conf`.

1. Open `ego.conf`.
    - Windows: `$EGO_CONFDIR\ego.conf`
    - Linux: `$EGO_CONFDIR/ego.conf`
2. Set the following parameter in the share directory:

   EGO_PARENT_QUOTA=**Y**
3. Save and close `ego.conf`.
4. Shutdown and then restart the cluster (vemkd must be restarted after configuring `ego.conf`).

Default resource allocation policy

When a consumer experiences demand, EGO considers resources from each of these five areas, and systematically allocates them (by default) in this order according to the configured resource plan:

1. Idle resources already owned by the consumer
2. Idle, unowned resources from the share pool
    1. Unowned resources within a particular branch in the consumer tree
    2. Unowned resources from other branches in the consumer tree, eventually moving up the tree to distribute any unowned resources from the cluster level
3. Idle resources owned by other consumers that are configured for lending (borrowed resources)
4. Resources owned by the consumer but currently lent-out to other consumers (reclaimed resources to owner)
5. Unowned resources from the share-pool but currently in use by consumers with a smaller share-ratio (reclaimed resources to share-pool)

You can change the default resource allocation policy so that owned resources get reclaimed by consumers before they are borrowed or allocated from elsewhere . You can also adjust the allocation policy so that resources are never reclaimed by the share pool, but are only returned when the borrowing client releases them.

The following table illustrates the revised order of resource allocation depending on configured policy changes.

Default allocation order	**Changed policy: Reclaim before borrow**	**Changed policy: No reclaim on share pool**
Idle resources owned by consumer	Idle resources owned by consumer	Idle resources owned by consumer
Idle, unowned resources from share pool	Idle, unowned resources from share pool	Idle, unowned resources from share pool
Idle resources borrowed from other consumers	Resources reclaimed by consumer	Idle resources borrowed from other consumers
Resources reclaimed by consumer	Resources reclaimed by share pool from over-allocated consumers and then re-allocated to deserving consumers with a higher-share ratio	Resources reclaimed by consumer
Resources reclaimed by share pool from over-allocated consumers and then re-allocated to deserving consumers with a higher-share ratio	Idle resources borrowed from other consumers	

Create or modify a resource plan

Make sure you are at the cluster or top level of your tree.

Note:

Before changing the default resource plan, you should export it. Doing this allows you to reset to the default plan, if required, by importing it again.

So that EGO allocates resources how and when you want them, create a resource plan to distribute resources from each resource group to each consumer. The more flexible your plan is, the more flexible your cluster is. For example, allowing lending and borrowing without setting limits helps your cluster react and change to the dynamic needs of your workload. However, you should not enable lending and borrowing for your InternalResourceGroup and ManagementHosts resource groups, nor change the default ownership; the services require a number of owned slots to run the cluster.

The Platform Management Console stores only your active resource plan. You can, however, import a resource plan in XML format to make updates or simply make changes to the resource plan from the Platform Management Console.

A simple default resource plan is already in place to adapt for your own purposes.

Any changes you make and apply to the resource plan in the Platform Management Console are implemented immediately. Do not apply the changes if you do not wish the changes to be immediate. Export the plan instead, and work on the copy until you are ready to import the updates and apply the changes.

1. Locate your resource plan by navigating to Consumers > Consumers & Plans > Resource Plan.

   Your resource plan displays.

   If you have never updated or imported a resource plan, your resource plan shows default settings.
2. Create and manage any time intervals.

   Select Time Intervals and Settings > Insert a time interval or Remove Time Interval: […], where […] is the name of a specific time interval.
3. If you have created resource groups, choose the resource group you want to set the resources for first from the drop down menu.

   You can insert different time intervals for different resource groups.

   The InternalResourceGroup and ManagementHosts resource groups are reserved for system services running your cluster. We do not recommend making changes to these two resource groups without an explicit Platform Support recommendation.
4. Click Expand All.

   Your entire consumer tree expands.
5. Select Time Intervals and Settings > Show Advanced Settings.

   Your advanced settings display, including lending and borrowing.
6. Set your owned slots for each consumer until the balance for each consumer branch in each time period is appropriate for your needs.

Under most circumstances, the balance should be zero. Under some circumstances (like host scavenging, if available), you may need to allocate more slots than you own.

Keep in mind that the resource plan is hierarchical in nature; you cannot allocate to a leaf consumer more resources than its branch owns.

7. (Optional, but recommended.) For each leaf consumer that has something registered to it, select the options to Lend and Borrow.

   Lending allows a consumer's unused slots to be used by other consumers. Borrowing lets a consumer use unowned or lent slots when they are available.

   Note:

   This step does not apply to the InternalResourceGroup or ManagementHosts resource groups that are reserved for running services that control your cluster. Do not make changes to the consumers that run system services. Do not enable lending or borrowing.

8. (Optional if Lend is checked) Next to Lend, click Details. In the Lend Details dialog box, specify the consumers you want to enable lending for and the maximum number of slots you would like to lend Limit field.

   This number is the maximum number of slots that can be lent out. The remainder are never lent, even if they are not being used. If you leave the field blank, all slots can be lent out.

   You can specify how many slots you want to lend to each individual consumer by clicking Details.

9. (Optional if Borrow is checked) Next to Borrow, specify the maximum number of slots you would like to borrow at any time in Limit.

   This number is the maximum number of slots that are ever borrowed by this consumer at any one time. If you leave the field blank, borrowing is limited only by the amount of resources available in the cluster.

   You can specify how many slots you want to borrow from each individual consumer by clicking Details. Select the consumer you want to enabling borrow for and use the Order column to specify the order in which you want to borrow from specific consumers, where 1 is first. Do not repeat any numbers.

   Note:

   If you modify the borrow order, you must run `egosh ego restart` from the command line.

10. (Optional) Specify the share ratio that applies across consumers at the same level in one branch.

    - If you want sibling consumers to share the resources equally, type 1 for all.
    - If you want one leaf consumer to have twice as many resources as its sibling, type 2 for the first consumer and 1 for the second consumer.
    - If you want one consumer to give up all its borrowed resources when a sibling has demand, specify 0 for the low-priority consumer.

    Your share ratio can be as simple or as complex as you like. Zero and all positive whole numbers are valid entries.

Note:

This step does not apply to the InternalResourceGroup or ManagementHosts resource groups that are reserved for running services that control your cluster. Do not make changes to the consumers that run system services; the share ratios should be uniform and there should be no limits.

11. Repeat the resource distribution for all time periods created.

    By default, there is only one time period (00:00 to 24:00).
12. Click Apply to save and make the current settings active.

    If you do not want to make the current changes active, export the resource plan instead.
13. Repeat the entire process for each resource group.

    Switch between resource groups using the drop down list above the plan. Make sure to apply changes before switching to a different resource group (click Apply).

Note:

The InternalResourceGroup and ManagementHosts resource groups are special. They are reserved for running system services that control the cluster. If you add your own services, you may need to allocate one or more slots from the ManagementHosts resource group to run workload. We do not recommend changing ownership or enabling lending or borrowing for consumers and services that are created for you by default.

You have modified the default resource plan and created a new plan that efficiently distributes resources across your cluster. Modifications have been applied and the plan is active. You can export this plan and import it again as needed if you want it to apply to specific days of the week only. You can create as many resource plans as you want and import them as needed for a quick change in your distribution.

Manage time intervals

You must have created at least one consumer before you can modify a time interval for a resource plan.

You must be a cluster administrator to manage time intervals.

Time intervals allow you to vary your resource plan according to the clock time of the day. For example, you may want to alter your distribution of resources overnight or in the evenings or both.

- Add a time interval
- Delete a time interval

Add a time interval

Make sure you are at the top of your tree (at the cluster level).

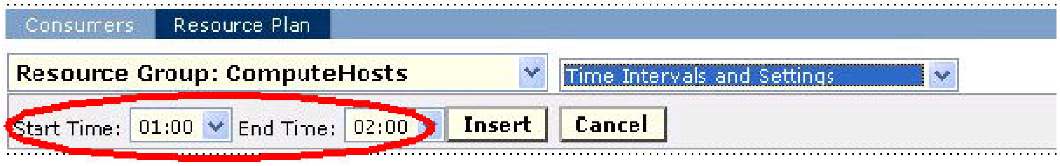
Time-based configuration in the resource plan allows the resource distribution for a consumer to change according to the time of day.

Time intervals exist across resource groups throughout the tree. No time gaps are allowed in the resource plan.

1. Click Consumers > Consumers & Plans.

   A list of consumers at that level in your tree displays.
2. Click Resource Plan.

   Your resource plan displays.
3. Select the resource group you want to add the time interval for.
4. Select Time intervals and settings > Insert a time interval.



Resource Group: ComputeHosts

Start Time: 01:00 End Time: 02:00

   An insert element displays just above the resource plan.
5. Specify a start time and an end time.

   By default, settings for a new time interval are inherited from an existing one.
6. Click Insert.
7. Click OK to confirm that you want to insert the time interval.

   The new time interval displays in your resource plan for the selected resource group.
8. Add the time interval for any other resource groups.

Delete a time interval

When you delete a time interval, the allocations you have assigned for the preceding time interval now apply to the whole time period. No time gaps are allowed in the resource plan.

You cannot delete the time interval 00:00 - 24:00 (leaving you with no time intervals); you can only modify it.

1. Go to your resource plan.
2. Select the resource group that has the time interval you want to remove.
3. Select Time intervals and settings > Remove Time Interval: [...] for the time interval you want.
4. Confirm that you want to remove that time interval.
5. Remove the time interval for any other resource groups.

Export a resource plan

You must be a cluster administrator or consumer administrator for a branch to export a resource plan.

Note:

Do not change your tree structure while the plan is exported or you cannot import the plan again.

Export a resource plan to modify the distribution of resources to your cluster offline, in an XML file. Use any XML editor to make changes.

1. Click Consumers > Consumers & Plans.

   A list of consumers at the top level of your tree displays.
2. Click Resource Plan.

   Your resource plan displays.
3. Go to the bottom of the page and click Export.
4. Save your resource plan locally.

   You may want to give the XML file a unique name including date or plan specifics so that you can import the plan of your choice quickly and easily at a later time.

Import a resource plan

You must be a cluster administrator or consumer administrator to import a resource plan.

The imported resource plan must have an identical tree structure to the existing resource plan at the time it is imported. However, time intervals may be different.

Import a modified resource plan (an XML file) to change your assignation of resources for your consumers.

Importing a resource plan makes it the active plan.

1. Click Consumers > Consumers & Plans.

   A list of any existing consumers at that level in your tree displays.
2. Click Resource Plan.

   If you do not see Resource Plan, make sure you are at the top of your tree.

   Your resource plan displays.
3. At the bottom of the page, click Import.

   The Import a resource plan (XML) dialog displays.
4. Browse for the location of the XML resource plan.
5. Click Import.

You have imported a resource plan and made it the active plan. Resource distribution takes effect immediately.

Change the default resource allocation policy in conjunction with resource plan

You must be a cluster administrator.

EGO systematically allocates resources according to a default plan. A key component of this plan is that consumers who experience demand borrow resources from other lending consumers before they reclaim any of their lent resources.

You can create a resource plan and change the resource allocation behavior so that owned resources get reclaimed by consumers before they are borrowed or allocated from elsewhere. This is done in coordination with an effective resource plan that properly reflects your business requirements.

1. Create an appropriate resource plan (Consumers > Consumers & Plans > Configure Resource Plan).

   If you have never created, updated, or imported a resource plan, the base plan is your only plan and it has default settings. See the topic Create or modify a resource plan for a detailed procedure. Find a summary below:

   a) Set the date and time as well as the frequency of occurrence.
   b) If you have created resource groups, choose the resource group you want to set the resources from the drop-down menu (for example, Resource Group: ComputeHosts).
   c) Select Show Advanced Settings.
   d) Click Expand All.
   e) Set your owned slots for each consumer until the balance for each consumer branch in each time period is appropriate for your needs.

      Under most circumstances, the balance should be zero. Under some circumstances (like host scavenging, if available), you may need to allocate more slots than you own.
   f) Rank your consumers, remembering that resources are reclaimed based on rank (those leaf consumers with a lower consumer rank are reclaimed before consumers with a higher rank).

      Specify any positive whole number, where 1 is the highest priority. Priority settings are relative to one another within the resource group. If you leave the priority blank, that consumer has no priority over any other consumer (it does not form part of any consumer ordering/sequencing).
   g) For each leaf consumer that has something registered to it, select the options to Lend and Borrow.

      Lending allows a consumer’s unused slots to be used by other consumers. Borrowing lets a consumer use unowned or lent slots when they are available.
   h) (Optional: if Lend is checked) Next to Lend, click Details. In the Lend Details dialog box, specify the maximum number of slots you would like to lend in the Total lend limit field.
   i) (Optional: if Borrow is checked) Next to Borrow, specify the maximum number of slots you would like to borrow at any time in Limit.
   j) (Optional) Specify the share ratio that applies across consumers at the same level in one branch.

      - If you want sibling consumers to share the resources equally, type 1 for all.

   - If you want one leaf consumer to have twice as many resources as its sibling, type 2 for the first consumer and 1 for the second consumer.
   - If you want one consumer to give up all its borrowed resources when a sibling has demand, specify 0 for the lower-priority consumer. Note that in doing this, the consumer does not receive any resource from the share pool.

   k) Click Apply to save and make the current settings active.

   If you do not want to make the current changes active, export the resource plan instead.

2. Set a reclaim grace period and rebalance behavior for a selected consumer:
   a) Click Consumers > Consumers & Plan.
   b) Click a top-level consumer from the consumer tree.
   c) Click Consumer Properties.
   d) Specify a reclaim grace period to apply when a resource gets reclaimed by its owner.
   e) Check the Rebalance when time intervals change box if you want EGO to “rebalance” or reset to the originally configured resource plan whenever a time interval change occurs (when there is a change in ownership of resources) or when resources are reclaimed (or passed back to their original owners).
   f) Click Apply.
3. Change the default reclaim behavior:
   a) Click Cluster > Summary > Cluster Properties.
   b) Specify the resource allocation behavior for the cluster by checking the appropriate boxes in the section Specify resource allocation behavior.
      - To allow the share pool to reclaim resources from an over-allocated consumer, ensure Reclaim shared resources is checked (default).
      - To allow a leaf consumer to reclaim its resources before borrowing from another consumer, check Reclaim lent resources before borrowing.
   c) Click Apply.

EGO continues to allocate resources according to your configured resource plan, but now considers systematically allocating resources in a different order.

Troubleshooting

Log files

When egosh finds errors while applying the resource plan, it shows detailed messages in stderr. When you specify an error log path, egosh appends error messages into the error log. This is useful when your resource plan is not a valid XML file, in which case the error message may include the whole XML string, which is a large amount of information to display as stderr.

The error log is named egosh.log.%hostname%.

CLI output

- If the resource plan is updated successfully, egosh shows the following messages:

  "The resource plan of consumer %s has been updated for DistributionTree %s"
- If the resource plan fails to update, egosh shows error messages for individual error cases. For example,

  "Your user account has insufficient rights to run the egosh command. Log on as Cluster Administrator or Consumer Administrator"
- If you run "egosh consumer applyresplan" with the "-c" option to check the applied resource plan and the resource plan is valid, egosh shows the following message:

  "The resource plan %s is valid; it is ok to apply the resource plan."

Feature: Resource plan modification by CLI feature

Use resource plan modification when you want to modify a resource plan using a script.

About resource plan modification by CLI

The EGO CLI Resource Plan Modification feature allows you to modify your resource plan using the command line interface (CLI). Use this feature when you want to modify a resource plan using a script. Benefits of using this feature include triggering the update of resource plan by means of a script. Without this feature, you must log on to the Platform Management Console (PMC) to modify the resource plan.

Scope

Applicability	**Details**
Operating system	Windows
EGO version	EGO 1.2.3
Commands	Applies only to the egosh command
Limitations	The feature is only available upon request
Known issues	None

Behavior of resource plan modification by CLI

After exporting, editing, and saving the resource plan as `C:\ResourcePlan.xml`, `egosh applyresplan C:\ResourcePlan.xml` updates the resource plan:

- Checks user rights.

  Only the Cluster Administrator or the Consumer Administrator of the top-level consumer can update the resource plan using egosh.
- Validates the applied resource plan against the schema.

  The applied resource plan must be a valid resource plan XML file.
- Checks for conflicts between the applied resource plan and the backend resource plan.

  The consumer in the applied resource plan must exist in the backend resource plan.
- Delivers the applied resource plan to the backend and makes it take effect.

  Calls the EGO API to deliver the applied resource plan to the backend.

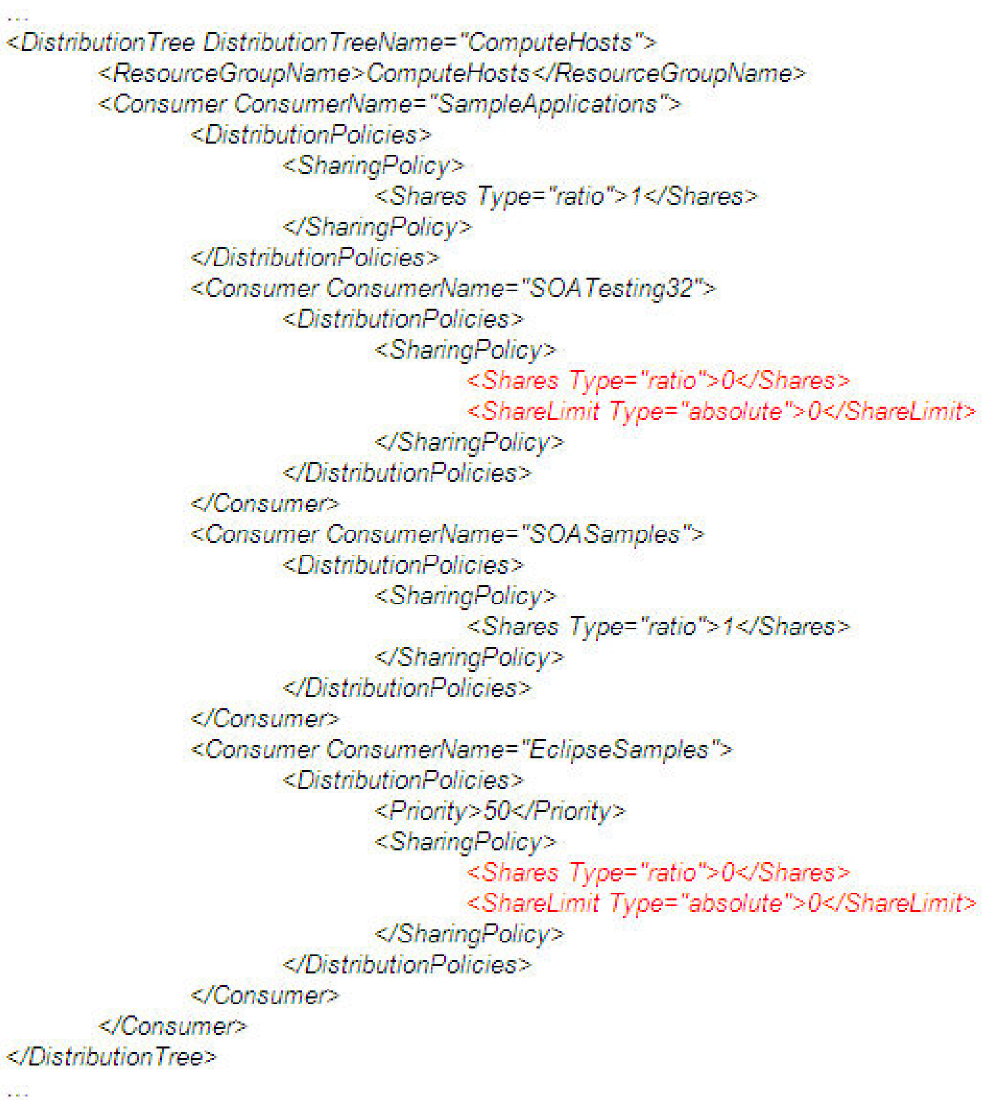
Usage

Syntax	**Description**
consumer applyresplan [-c] [-e *error_log_directory*] *file_path*	Applies the specified resource plan
-c	Only checks the resource plan without applying it.

Syntax	Description
-e	Specifies the directory of the error log. If an error occurs while updating the resource plan, the error messages are appended into the error log in the specified directory. If the directory of the error log is not specified, the system outputs error messages to `stderr` .
file_path	Specifies the path of the resource plan that you want to apply. The resource plan must be a valid resource plan XML file.

Example of usage

1. Prepare the resource plans.
    1. Log on to the Platform Management Console as a cluster administrator.
    2. Export the resource plan and save it as `C:\ResourcePlan.xml`.
    3. Edit ResourcePlan.xml in order to allocate all the resources available to SOASamples, and save it as `C:\ResourcePlan_new.xml`.



```
...
<DistributionTree DistributionTreeName="ComputeHosts">
        <ResourceGroupName>ComputeHosts</ResourceGroupName>
        <Consumer ConsumerName="SampleApplications">
                <DistributionPolicies>
                        <SharingPolicy>
                                <Shares Type="ratio">1</Shares>
                        </SharingPolicy>
                </DistributionPolicies>
                <Consumer ConsumerName="SOATesting32">
                        <DistributionPolicies>
                                <SharingPolicy>
                                        <Shares Type="ratio">0</Shares>
                                        <ShareLimit Type="absolute">0</ShareLimit>
                                </SharingPolicy>
                        </DistributionPolicies>
                </Consumer>
                <Consumer ConsumerName="SOASamples">
                        <DistributionPolicies>
                                <SharingPolicy>
                                        <Shares Type="ratio">1</Shares>
                                </SharingPolicy>
                        </DistributionPolicies>
                </Consumer>
                <Consumer ConsumerName="EclipseSamples">
                        <DistributionPolicies>
                                <Priority>50</Priority>
                                <SharingPolicy>
                                        <Shares Type="ratio">0</Shares>
                                        <ShareLimit Type="absolute">0</ShareLimit>
                                </SharingPolicy>
                        </DistributionPolicies>
                </Consumer>
        </Consumer>
</DistributionTree>
...
```

2. Switch resource plans using egosh in your script.
    1. Log on as a cluster administrator in egosh.

       The credentials are saved in a temp directory. Over the next 8 hours, egosh uses this credential to execute commands and does not need the user's password.
    2. In your script, call `egosh applyconsumer -e C:\ C:\ResourcePlan.xml` and `egosh applyconsumer -e C:\ C:\ResourcePlan_new.xml` to switch resource plans when some event occurs.

Behavior of configuration to modify resource plan modification by CLI

None.

Resource plan modification by CLI commands

Commands to monitor

User	Command	Behavior
Cluster Administrator or Consumer Administrator	From the Platform Management Console: **Consumers & Plans** > **Resource Plan**	Check the resource plan.

Commands to control

User	Command	Behavior
Cluster Administrator or Consumer Administrator	From the Platform Management Console: **Consumers & Plans** > **Resource Plan**	Export the resource plan.
Cluster Administrator or Consumer Administrator	`egosh consumer applyresplan`	Modify the resource plan.

Commands to display configuration

Not applicable.

Commands for submission

Syntax	Description
consumer applyresplan [-c] [-e *error_log_directory*] *file_path*	Applies the specified resource plan
-c	Only checks the resource plan without applying it.
-e	Specifies the directory of the error log. If an error occurs while updating the resource plan, the error messages are appended into the error log in the specified directory. If the directory of the error log is not specified, the system outputs error messages to stderr.
file_path	Specifies the path of the resource plan that you want to apply. The resource plan must be a valid resource plan XML file.

PART

III

Inside Platform EGO: Advanced Topics and Troubleshooting

CHAPTER

16

Advanced Concepts and Overviews

EGO components

The components of the resource management middleware layer (for the supply side) fall into several groupings. Those that play a more central and exposed role include the cluster kernel, system services, APIs, and application orchestrators. Each group performs a set of functions essential to orchestrating the access of applications to the underlying resources.

Name	**Description**
EGO cluster kernel	The enterprise cluster kernel is a process automatically started on one of the hosts within the cluster. It provides a core set of centralized functions that leverage the capabilities of host-level agents. The kernel integrates the resources of all hosts and performs many-to-one virtualization so that mixed physical resources appear to clients as a single virtual computer. It is in the EGO cluster kernel where the requirements for support for services and for plan-based ownership are primarily met. There are three parts that makeup the EGO cluster kernel: Information, Allocation, and Execution.
Execution	Once resources are allocated, the EGO cluster kernel provides the mechanism that allows them to execute activities using those resources. Actions include starting, stopping, or controlling execution activities. The EGO cluster kernel uses process execution managers (PEMs) on the hosts to perform remote operations using operating system-level processes. Changes of status on the host are reported asynchronously to the client so that the client can determine how to handle failures and restarts.
Allocation	The EGO cluster kernel also manages allocation requests from clients. Like a virtual memory manager in a host operating system, which converts a physical resource into a virtual resource and allocates it to applications, the EGO cluster kernel virtualizes distributed resources. For example, it can take physical hosts and carve them up into virtual CPU slots. Clients could then request and release CPU slots through allocate and release interfaces, identifying the number and type of resources they need based on host attributes of allocating consumers. Taking into consideration the available resources and consumer entitlements, the EGO cluster kernel applies plans to determine what to allocate. The client is notified asynchronously as resources become available, the resources are identified, and host information is passed back to the client. The allocation engine balances the demand for resources with the available supply. It tracks the amount of each type of resource and adjusts priorities, reclaiming resources from clients that exceed their share of the resources. The allocation engine supports immediate allocation requests.
Information	The EGO cluster kernel aggregates information from the hosts, providing a single point from which clients can request information such as the state of individual resources, the status of allocation requests, the consumer hierarchy, including resources assigned to each consumer, and services that have been started. By giving central access to static and near-real-time information, the kernel makes it possible to effectively monitor and manage resources and enables clients to discover what resources are available.

Name	Description
System services	System services provide common functionality required to support multiple workload-specific application managers. Higher-level services leverage these common services to ensure consistent management of distributed application workloads. There are a number of system services, including the service controller, WebServiceGateway, service director (ServiceDirector), and Platform Management Console (WEBGUI) services. These are described below.
Service controller	The service controller is always started on the EGO cluster kernel host. It is responsible for starting other system services. Acting as a client to the EGO cluster kernel, it requests resource allocations for running services. It ensures that services are running by detecting failures and restarting service instances based on availability plans.
ServiceDirector	Services are dynamically started and can therefore run on any host. This means that clients have to dynamically locate services to access them. Clients can locate services explicitly via API calls to the service controller that starts and tracks those services; alternatively, the service director provides a mechanism, based on standard DNS, for redirecting client requests to the physical instance of that service.
WEBGUI	The WEBGUI service (Platform Management Console) is a web-based graphical user interface for administrators and operations staff. The Console is extensible to enable higher-order services to add functionality for managing workload- or application-specific interfaces.
WebServiceGateway	The web service gateway (WebServiceGateway) service is a runtime component of EGO. The gateway provides a standards-based web services interface for web service clients to access EGO functionality. The web service client sends its request to the gateway via SOAP protocol. The gateway calls the EGO C APIs to perform the required operations on behalf of the web service client and returns the results.
RS	The repository service manages package deployment, and allows deployment of a service without reliance on a shared file system. When a service package is updated, the new service package overwrites the existing service package. The repository service is started as a EGO system service. If the process or host fails, the service is relocated and restarted on another host.
purger	The purger service is used by the reporting feature, and manages the relational database size by deleting old data at regular intervals.
plc	The plc service is used by the reporting feature, and manages the data loader plug-ins.
derbydb	This Apache Derby service runs as a system service when first installed. It is used primarily for demo clusters. The derbydb service is only enabled if an environment variable is set prior to installation (Linux) or during the Windows installation.
EGO cluster interfaces	Standards-based EGO cluster interfaces help meet the requirement of being application agnostic and facilitate integration with heterogeneous applications, enabling third-party vendors and customers to easily integrate applications and resources across the cluster.

Name	Description
EGO application orchestrators and applications	Another key factor in meeting the requirement to support services is EGO Application Orchestrators and Applications-higher-order services that provide workload or application-specific integration with the cluster. A EGO Application Orchestrator understands the nature of an application workload, measures its performance against service levels, and pinpoints bottlenecks. If the orchestrator determines that additional resources can alleviate a performance problem, it leverages other components to request resources and then release them when they are no longer needed. EGO Application Orchestrators can also process workloads and act as application middleware, providing EGO interfaces for users to develop applications.

System services and EGO daemons

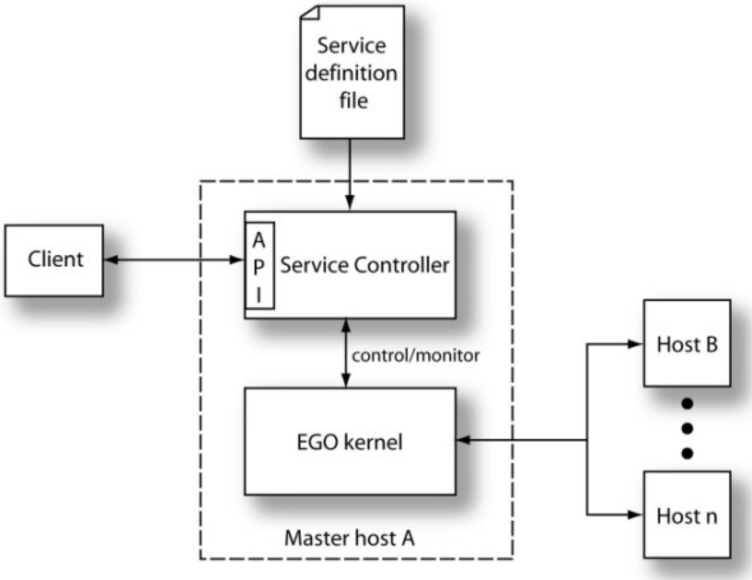
A system service is a self-contained, continuously running process that accepts one or more requests and returns one or more responses. Services may have multiple concurrent service instances running on multiple hosts. Most system services are automatically enabled by default at installation (derbydb is the exception, and must be manually enabled during installation).

About the service controller

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 defined services are running by detecting failures and restarting service instances based on the parameter settings in the Control Policy portion of the service profile.

The service controller also provides APIs to allow other tools to instantiate, control, and query services at runtime.



Service definition

The service controller reads configuration files for services that must be instantiated. The configuration files are XML documents (service profile) that contain the service definition (that is, parameters that define the type of resources the service instances need to run along with how to start and monitor the service instances). These files can be created either by the service controller administrator or by the API during runtime. There is one service definition file dedicated to each service.

Start-up sequence

Service controller start-up sequence:

1. One of the hosts within the cluster is elected as the master. The master host, in turn, starts the EGO kernel, which then starts the service controller on the same host.
2. The service controller opens a connection to EGO and registers as a recoverable client.
3. The service controller loads the service definition database containing the list of services that need to be instantiated.
4. The service controller recovers the latest state info from EGO and starts and stops services following the service definitions from the database. The service controller is ready for service.

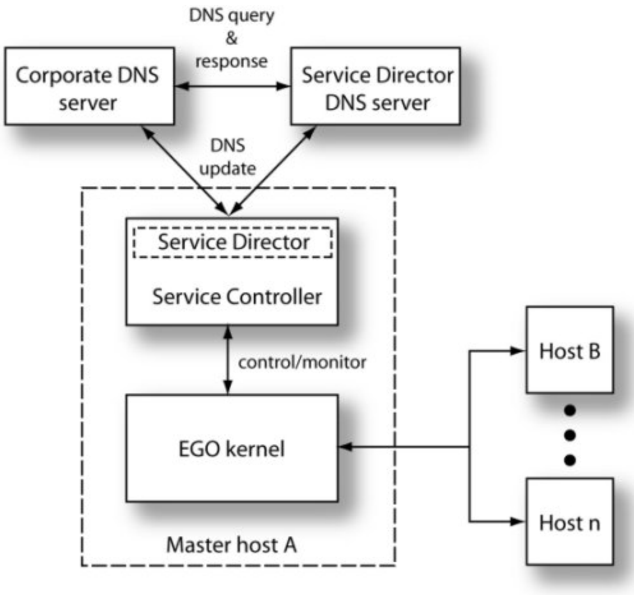
About the service director

The service director is a basic system service that functions as a locating mechanism for other system services. The service director contains a stand-alone Domain Name Server (DNS), which is the authoritative name server for the EGO DNS sub-domain and responds to DNS queries for system services.

The service director runs on the EGO master host and relies on the service controller to provide location information and state change notifications of service instances.

When a service instance enters the RUN state, the service director adds its location information into the service director DNS server. When a service instance transfers from the RUN state into other states, the service director deletes the location information from its DNS server.

When the locations of the service director DNS server or other system services are changed, the service director updates the corresponding resource record in the DNS database of the corporation DNS server or service director DNS server, respectively.



Service director DNS server

The service director DNS server is a stand-alone DNS server and has the responsibility to process DNS queries and maintain the mapping between server names and IP addresses. There is only one service director DNS server running in the EGO environment.

Service director start-up and recovery

The service director DNS server is essential to the operation of the service director. Consequently, the service director DNS server is configured with automatic startup, and therefore, the service controller starts the service director DNS server whenever it finds the DNS server is not running.

When the service controller restarts, it recovers the information of all the services from the EGO kernel. If the service controller finds the service director DNS server is still running, it recovers it.

During recovery, the service director performs the following steps:

1. Updates the service director DNS server location information

   Removes the old location information of the service director DNS server from the corporation DNS server, and then adds the new location information.
2. Updates services location information

   Removes the old location information of all current services, and then adds the current location information of services that have running instances.

Service instance location update

When a service instance is down and there is no other instance running on the same host, the service director deletes its location information from the service director DNS server.When a service instance is running, the service director adds its location information in the service director DNS server. The service director DNS server handles the duplicate location information.

About WEBGUI service and Web server

The WEBGUI service provides a high level view of running system services from the Platform Management Console. A cluster administrator can view detailed system service information and assess whether any actions are required for system services and service instances.

Service configuration can be done via the Platform Management Console. Configurations done through the Console are updated automatically in a service profile (XML file). The system service is also registered through the Platform Management Console.

The Web server is the host that runs the WEBGUI service (in effect, the Platform Management Console). Only one management host is elected as the Web server host.

About WebServiceGateway

The web service gateway (WebServiceGateway) service is a runtime component of EGO. The gateway provides a standards-based web services interface for web service clients to access EGO functionality. The web service client sends its request to the gateway via SOAP protocol. The gateway calls the EGO C APIs to perform the required operations on behalf of the web service client and returns the results.

Daemon file names

The following table outlines the EGO daemons and their associated log file names. Log files on Windows hosts have a `.txt` extension. Audit logs must be enabled first.

Daemon	**Log file name**
tomcat (WEBGUI)	`catalina.out`
datasourcetools (Database Configuration Tool)*	`datasourcetools.`*`hostname`*`.log`
egoconsumerresloader (Consumer Resource Data Loader)*	`egoconsumerresloader.`*`hostname`*`.log`
egodynamicresloader (Dynamic Metric Data Loader)*	`egodynamicresloader.`*`hostname`*`.log`
egoeventsloader (EGO Events Data Loader)*	`egoeventsloader.`*`hostname`*`.log`
egosc (EGO Service Controller)	`egoservice.audit.log`, `esc.log.`*`hostname`*
egostatisticresloader (Static Attribute Data Loader)*	`egostatisticresloader.`*`hostname`*`.log`
fam (File Access Manager)	`fam.`*`hostname`*`.log`
lim (Load Information Manager)	`lim.log.`*`hostname`*
named (Service Director)	`named.log`
pem (Process Execution Manager)	`pem.log.`*`hostname`*
pim (Process Information Manager)	`pim.log.`*`hostname`* (Linux only)
plc (Loader Controller)*	`plc.`*`hostname`*`.log`
purger (Data Purger)*	`purger.`*`hostname`*`.log`
rfa (Remote File Access)	`cli.`*`hostname`*`.log`
rs (Repository Service)	`rs.`*`hostname`*`.log`, `repositoryservice.audit.log`
vemkd (EGO Kernel Daemon)	`ego.audit.log`, `vemkd.log.`*`hostname`*
WSG (Web Service Gateway)	`wsg.log`
WSM (Platform Management Console/WEBGUI)	`wsm.log.`*`hostname`*

Understanding how lim determines host models and types

The lim (load information manager) daemon/service automatically collects information about hosts in an EGO cluster, and accurately determines running host models and types. At most, 1024 model types can be manually defined in `ego.shared`:

- Windows: `$EGO_CONFDIR\ego.shared`
- Linux: `$EGO_CONFDIR/ego.shared`

If `ego.shared` is not fully defined with all known host models and types found in the EGO cluster, the lim attempts to match an unrecognized running host to one of the models and types that is defined.

The lim supports both exact matching of host models and types, and "fuzzy" matching (where an entered host model name or type is slightly different from what is defined in `ego.conf`).

How does "fuzzy" matching work?

The lim reads host models and types that have been manually configured in `ego.shared`. The format for entering host models and types in `ego.shared` is *model_bogomips_architecture* (for example, **x15_4604_OpterontmProcessor142**, **IA64_2793**, or **SUNWUltra510_360_sparc**). Names can be up to 64 characters long.

When the lim attempts to match running host model with what's entered in `ego.shared`, it first attempts an exact match, then proceeds to make a fuzzy match. Here is a summary on how the lim attempts to make matches, depending on given information:

Architecture name of running host	What the lim reports	Additional information about the lim process
Same as definition in `ego.shared` (exact match)	Reports the reference index of exact match	The lim detects an exact match between model and input architecture string

Architecture name of running host	What the lim reports	Additional information about the lim process
Similar to what is defined in `ego.shared` (fuzzy match)	Reports fuzzy match based on detection of 1or 2 fields in the input architecture string	• For input architecture strings with only one field: If the lim cannot detect an exact match for the input string, then the lim reports the best match; a "best match" is defined as a model field with the most characters shared by the input string
		• For input architecture strings with two fields: **1.** If the lim cannot detect an exact match, it attempts to find a best match by identifying the *model* field with the most characters that match the input string **2.** The lim then attempts to find the best match on the *bogomips* field
		• For architecture strings with three fields: **1.** If the lim cannot detect an exact match, it attempts to find a best match by identifying the *model* field with the most characters that match the input string **2.** After finding the best match for the model field, the lim attempts to find the best match on the *architecture* field **3.** The lim then attempts to find the closest match on the *bogomips* field, with wildcards supported (where the *bogomips* field is a wildcard)
Has an illegal name	Reports default host model	An illegal name is one that does not follow the permitted format for entering an architecture string where the first character of the string is not an English-language character.

Safely making configuration file changes

It is important and highly recommended to configure EGO through the Platform Management Console, not directly within the various configuration files.

Be aware of cause-effect relationships that exist in EGO between files. For example, manual changes made to the `ego.conf`, `ResourceGroup.xml` or `ConsumerTrees.xml` files may potentially affect other EGO configuration settings.

Furthermore, to achieve a specific goal, a whole set of related parameters must often be changed or tuned collectively, some in obscure directory locations. If not done correctly, the output of manually updated configuration files may not produce the expected behavior.

Finally, with certain configuration settings, important validations triggered through the Platform Management Console must be conducted by EGO. By using the Platform Management Console, legitimate and allowable parameter settings are ensured.

Examples of potential problems caused by manually configuring files:

- If you manually configure a consumer policy within “DistributionTree” sections of the `ConsumerTrees.xml` file without adding a corresponding instance in the “ConsumerHierarchy” section, then EGO does not recognize the newly added consumer.
- If you manually configure borrow and lend plans without giving a full consumer path (including the consumer tree name), then EGO may ignore them.
- If you set an invalid time window (not covering a 0 to 24 hour time period), then resource plans do not behave as expected.
- If you allot an unbalanced ownership, (such as reconfiguring a leaf consumer’s ownership value without changing the value at the branch level), then resource plans are not be effective and workload units do not run as expected.
- If you delete a consumer manually without checking whether it owns allocations or is currently running activities, you can seriously affect your resource plan and running workload units, among other things.

Events overview

An event is an occurrence of something significant happening in the cluster. Notification of events is an important tool because it allows the administrator to be informed about and to monitor the health of the cluster.

Events have three levels: LOG_ERR, LOG_WARNING, and LOG_INFO.

- LOG_ERR: Error events only
- LOG_WARNING: Warning and error events
- LOG_INFO: All events

Each event has the following fields:

- Component: String that identifies that component triggered the event
- Code: An event number used for identification
- Name: The name of the event
- Level: The event level this event belongs to (for example, LOG_WARNING)
- Args: Arguments to the event

Note:

Together, component and code provide unique event identification.

By default, the event feature is not enabled.

Default SNMP

EGO includes an SNMP plug-in: `eventplugin_snmp`. This plug-in is integrated with SNMP, which uses SNMP traps as the notification mechanism. The file is located under $EGO_LIBDIR in the file `eventplugin_snmp.dll`. On a Linux system, the file has the extension `.so`.

The default configuration of this plug-in is in `ego.conf` under the section `EGO event configuration`. It is set as follows:

```
EGO_EVENT_MASK=LOG_INFO
EGO_EVENT_PLUGIN=eventplugin_snmp[SINK=host,MIBDIRS=EGO_CONFDIR/mibs]
```

(where *host* is the host that the SNMP trap is sent to.)

Note:

The `MIBDIRS` directory may also equal `%EGO_CONFDIR%/kernel/conf/mibs`.

In a Windows environment, use quotation marks around the event plug-in definition. For example:

```
EGO_EVENT_PLUGIN="eventplugin_snmp
[SINK=host,MIBDIRS=$EGO_CONFDIR\kernel\conf
\mibs]"
```

By default, all SNMP traps are sent to port 162. To change the destination port, an optional string "`TRAPPORT=`*`port_number`*" may be added to the `EGO_EVENT_PLUGIN` configuration string above.

The definition of the SNMP trap sent by this plug-in is defined in the file `EGO-SNMP-MIB.txt` found under *`EGO_CONFDIR`*`/mibs`.

You can replace the default SNMP with another SNMP. We recommend SNMP Trap Watcher by BTT Software.

CHAPTER

17

Logs and Traces

About EGO log files

Log files contain important run-time information about the general health of EGO daemons and EGO system events. Log files are an essential troubleshooting tool during production and testing.

The naming convention for most EGO log files is the name of the daemon plus the host name the daemon is running on.

The following table outlines the EGO daemons and their associated log file names. Most log files on Windows hosts have a `.txt` extension.

Daemon or component	Log file name
Application configuration wizard	In the Platform Management Console, **Reports** > **Standard Reports**
datasourcetools (Database Configuration Tool)*	`datasourcetools.`*`host_name`*`.log`
egoconsumerresloader (Consumer Resource Data Loader)*	`egoconsumerresloader.`*`host_name`*`.log`
egodynamicresloader (Dynamic Metric Data Loader)*	`egodynamicresloader.`*`host_name`*`.log`
egoeventsloader (EGO Events Data Loader)*	`egoeventsloader.`*`host_name`*`.log`
egosc (EGO Service Controller)	`egoservice.audit.log`, `esc.log.`*`host_name`*
egostatisticresloader (Static Attribute Data Loader)*	`egostatisticresloader.`*`host_name`*`.log`
fam (File Access Manager)	`fam.`*`host_name`*`.log`
lim (Load Information Manager)	`lim.log.`*`host_name`*
named (Service Director)	`named.log`
pem (Process Execution Manager)	`pem.log.`*`host_name`*
pim (Process Information Manager)	`pim.log.`*`host_name`* (Linux only)
plc (Loader Controller)*	`plc.`*`host_name`*`.log`
purger (Data Purger)*	`purger.`*`host_name`*`.log`
rfa (Remote File Access)	`cli.`*`host_name`*`.log`
rs (Repository Service)	`rs.`*`host_name`*`.log`, `repositoryservice.audit.log`
tomcat (WEBGUI)	`catalina.out`
vemkd (EGO Kernel Daemon)	`ego.audit.log`, `vemkd.log.`*`host_name`*
WSG (Web Service Gateway)	`wsg.log`

Daemon or component	Log file name
WSM (Platform Management Console/WEBGUI)	`wsm.log.`*`host_name`*

The majority of log entries are informational in nature. It is not uncommon to have a large (and growing) log file and still have a healthy cluster.

*Indicates daemons and log files associated with the reporting feature.

Log file locations

By default, most EGO log files are found in *`EGO_TOP`*`\kernel\log` (Windows) or *`EGO_TOP`*`/kernel/log` (Linux).

Component	Log file name	Windows	Linux
Application configuration wizard	• `upgradeLog.log`	In the Platoform management console: **Logs** > **Standard Logs**	
EGO kernel daemon	• `ego.audit.log` • `vemkd.log.`*`host_name`*	*`EGO_TOP`*`\kernel\log`	*`EGO_TOP`*`/kernel/log`
EGO service controller	• `egoservice.audit.log`, `esc.log.`*`host_name`*	*`EGO_TOP`*`\eservice \esc\log`	*`EGO_TOP`*`/eservice/ esc/log`
File access manager	• `fam.`*`host_name`*`.log`	*`EGO_TOP`*`\kernel\log`	*`EGO_TOP`*`/kernel/log`
Load information manager	• `lim.log.`*`host_name`*	*`EGO_TOP`*`\kernel\log`	*`EGO_TOP`*`/kernel/log`
Process execution manager	• `pem.log.`*`host_name`*	*`EGO_TOP`*`\kernel\log`	*`EGO_TOP`*`/kernel/log`
Process information manager	• `pim.log.`*`host_name`*	N/A	*`EGO_TOP`*`/kernel/log`
Platform management console/WEBGUI	• `catalina.out` • `wsm.log.`*`host_name`*	*`EGO_TOP`*`\gui\logs`	*`EGO_TOP`*`/gui/logs`
Remote file access: client side	• `cli.`*`host_name`*`.log`	Where `rfa` was run from	Where `rfa` was run from
Remote file access: server side	• `cli.`*`host_name`*`.log`	*`EGO_TOP`*`\eservice \rs\log`	*`EGO_TOP`*`/eservice/ rs/log`

Component	Log file name	Windows	Linux
Reporting: - Database configuration tool - Dynamic metric data loader - EGO events data loader - Static attribute data loader - Loader controller - Data purger	- `datasourcetools.`*`host_name`*`.log` - `egodynamicresloader.`*`host_name`*`.log` - `egoeventsloader.`*`host_name`*`.log` - `egostatisticresloader.`*`host_name`*`.log` - `plc.`*`host_name`*`.log` - `purger.`*`host_name`*`.log`	*`EGO_TOP`*`\perf\logs`	*`EGO_TOP`*`/perf/logs`
Repository service and file access manager	- fam.*host_name*.log - `rs.`*`host_name`*`.log` - `repositoryservice.audit.log`	*`EGO_TOP`*`\eservice\rs\log`	*`EGO_TOP`*`/eservice/rs/log`
Service director	- `named.log`	*`EGO_TOP`*`\eservice\esd\conf\named\namedb`	*`EGO_TOP`*`/eservice/esd/conf/named/namedb`
Web service gateway	- `wsg.log`	*`EGO_TOP`*`\eservice\wsg\log`	*`EGO_TOP`*`/eservice/wsg/log`

Log files can also be accessed through the Platform Management Console (from System Logs > Standard Logs).

Log entry format

The standard format for log file entries is:

date time_zone log_level [*process_id*:*thread_id*] *action:description/message*

where the date is expressed in YYYY-MM-DD hh:mm:ss.sss.

For example, `2006-03-14 11:02:44.000 Eastern Standard Time ERROR [2488:1036] vemkdexit: vemkd is halting`.

Log classes for vemkd and pem

Use the following parameters to specify the log class:

- vemkd: EGO_DEBUG_VEMKD

  For example, `EGO_DEBUG_VEMKD=LC_AUTH`.
- pem: EGO_DEBUG_PEM

  For example, `EGO_DEBUG_PEM=LC_PEM`

Every log entry belongs to a log class. You can use log class as a mechanism to filter log entries by area. Log classes in combination with log levels allow you to troubleshoot using log entries that only address, for example, configuration.

Log classes (as well as log levels) can be filtered at run time using `egosh debug`.

Valid logging classes are as follows:

Log class	Description
LC_TRACE	Logs significant program steps.
LC_COMM	Logs messages related to communications.
LC_AUTH	Logs messages related to users and authentication.
LC_MEM	Logs messages related to memory allocation.
LC_SYS	Logs messages related to system calls.
LC_PERF	Logs messages related to performance.
LC_RSRC	Logs messages related to resources, including host status changes.
LC_ALLOC	Logs messages related to the resource allocation engine.
LC_ACTIVITY	Logs messages related to activities.
LC_PEM	Logs messages related to the process execution manager (pem).
LC_EVENT	Logs messages related to the event notification service.
LC_QUERY	Logs messages related to client queries.
LC_RECOVER	Logs messages related to recovery and data persistence
LC_CONF	Logs messages related to configuration.
LC_CLIENT	Logs messages related to clients.

Log classes for lim

Use EGO_DEBUG_LIM to specify the log class. For example, `EGO_DEBUG_LIM=LC_MEMORY`.

Every log entry belongs to a log class. You can use log class as a mechanism to filter log entries by area. Log classes in combination with log levels allow you to troubleshoot using log entries that only address, for example, configuration.

Log classes (as well as log levels) can be filtered at run time using `egosh debug`.

Valid logging classes are as follows:

Log class	Description
LC_SCHED	Logs LSF scheduler (mbschd) messages.
LC_EXEC	Logs significant steps for job execution.
LC_TRACE	Logs significant program steps.
LC_COMM	Logs messages related to communications.
LC_XDR	Logs everything transferred by XDR

Log class	Description
LC_CHKPNT	Logs checkpointing messages.
LC_LICENSE	Logs license management messages.
LC_FILE	Logs file transfer messages.
LC_AFS	Logs AFS messages.
LC_AUTH	Logs messages related to users and authentication.
LC_HANG	Marks where a program might hang.
LC_MULTI	Logs messages pertaining to MultiCluster.
LC_SIGNAL	Logs messages pertaining to signals.
LC_DCE	Logs messages pertaining to DCE support.
LC_PIM	Logs PIM messages.
LC_MEMORY	Logs memory limit messages.
LC_SYS	Logs system call messages.
LC_JLIMIT	Logs job slot limit messages.
LC_FAIR	Logs fairshare policy messages.
LC_PREEMPT	Logs preemption policy messages.
LC_PEND	Logs messages related to job pending reasons.
LC_EEVENTD	Logs eeventd messages.
LC_LOADINDX	Logs load index messages.
LC_RESOURCE	Logs information used by resource broker (resource gathering and reporting).
LC_JGRP	Logs job group messages.
LC_JARRAY	Logs job array messages.
LC_MPI	Logs MPI messages.
LC_ELIM	Logs ELIM messages.
LC_M_LOG	Logs multievent logging messages.
LC_PERFM	Logs performance messages.
LC_HPC	Logs information specific to HPC integration.
LC_LICSCHED	Logs LSF License Scheduler messages.

Log levels

Use EGO_LOG_MASK to specify the log level. For example, `EGO_LOG_MASK=LOG_CRIT`.

For most logs, there are nine log levels that allow administrators to control the level of event information that is logged. For logs associated with the reporting feature, there are seven log levels.

When you are troubleshooting, increase the log level to obtain as much detailed information as you can. When you are finished troubleshooting, decrease the log level to prevent the log files from becoming too large and to enhance daemon performance.

Valid logging levels are as follows (not including the reporting feature log levels):

Log level	Description
LOG_EMERG	Logs only those messages in which the system is unusable.
LOG_ALERT	Logs those messages for which action must be taken immediately.
LOG_CRIT	Logs those messages that are critical.
LOG_ERR	Logs those messages that indicate error conditions.
LOG_WARNING	Logs those messages that are warnings or more serious messages. This is the default level of debug information.
LOG_NOTICE	Logs those messages that indicate normal but significant conditions or warnings and more serious messages.
LOG_INFO	Logs all informational messages and more serious messages.
LOG_DEBUG	Logs all debug-level messages.
LOG_TRACE	Logs all available messages. **Note:** LOG_TRACE is not supported by the LIM. If you set LOG_TRACE for the LIM, it is automatically changed to LOG_DEBUG.

Valid log levels for reporting feature are as follows:

Log level	Description
OFF	Logs no messages.
FATAL	Logs messages that were fatal to the reporting feature.
ERROR	Logs those messages that indicate error conditions.
WARN	Logs those messages that are warnings or more serious messages.
INFO	Logs all informational messages and more serious messages. (Default)
DEBUG	Logs all debug-level messages
ALL	Logs all messages.

Conf files where log level and class information are retrieved

The lim, pem, and vemkd daemons read `ego.conf` to retrieve the following information (as corresponds to the particular daemon).

- EGO_LOG_MASK: The log level used to determine the amount of detail logged.
- EGO_DEBUG_LIM: The log class setting for lim.
- EGO_DEBUG_PEM: The log class setting for pem.
- EGO_DEBUG_VEMKD: The log class setting for vemkd.

Fastpath:

- Windows: `%EGO_CONFDIR%\kernel\conf\ego.conf`
- Linux: `$EGO_CONFDIR/kernel/conf/ego.conf`

The service director daemon ("named") reads `named.conf` to retrieve the following information:

- logging severity: The configured severity log class controlling the level of event information that is logged (critical, error, warning, notice, info, debug, or dynamic). In the case of the log class set to debug, a log level is required to determine the amount of detail logged for debugging purposes. The higher the log level number, the more debug details messages are logged. Refer to third-party documentation for more information about BIND and logging.

Fastpath:

- Windows: `EGO_TOP\eservice\esd\conf\named\conf\named.conf`
- Linux: `EGO_TOP/eservice/esd/conf/named/conf/named.conf`

The egosc daemon reads `egosc_conf.xml`.

Fastpath:

- Windows: `EGO_TOP\eservice\esc\egosc_conf.xml`
- Linux: `EGO_TOP/eservice/esc/egosc_conf.xml`

The wsg daemon reads `wsg.conf` to retrieve the following information:

- WSG_DEBUG_DETAIL: The log level used to determine the amount of detail logged for debugging purposes. The configured severity log class controlling the level of event information that is logged (critical, error, warning, notice, info, debug, or dynamic). In the case of the log class set to debug, the logging is either on (1) or off (0).
- WSG_LOGDIR: Where to write `wsg.log` files.

Fastpath:

- Windows: `%EGO_CONFDIR%\kernel\conf\wsg.conf`
- Linux: `$EGO_CONFDIR/kernel/conf/wsg.conf`

The wsm daemon reads `wsm.conf` to retrieve the following information:

- LOG_LEVEL: The configured log level controlling the level of event information that is logged (INFO, ERROR, WARNING, or DEBUG).

Fastpath:

- Windows: *EGO_TOP*`\gui\conf\wsm.conf`
- Linux: *EGO_TOP*`/gui/conf/wsm.conf`

If a system is running well, typically set log level to info or even warning to minimize messages.

Note:

The daemons associated with the reporting feature read various `.xml` files to retrieve information. For more information, see the Reports chapters.

Why do log files grow so quickly?

Every time an EGO system event occurs, a log file entry is added to a log file. Most entries are informational in nature, except when there is an error condition. If your log levels provide entries for all information (for example, if you have set them to LOG_DEBUG), the files grow quickly.

Suggested settings:

- During regular EGO operation, set your log levels to LOG_WARNING. With this setting, critical errors are logged but informational entries are not, keeping the log file size to a minimum.
- For troubleshooting purposes, set your log level to LOG_DEBUG. Because of the quantity of messages you receive when subscribed to this log level, change the level back to LOG_WARNING as soon as you are finished troubleshooting.

Note:

If your log files are too long, you can always rename them for archive purposes. New, fresh log files are then created and log all new events.

How often should I maintain log files?

The growth rate of the log files is dependent on the log level and the complexity of your cluster. If you have a large cluster, daily log file maintenance may be required.

We recommend using a log file rotation utility to do unattended maintenance of your log files. Failure to do timely maintenance could result in a full file system, which hinders system performance and operation.

Change EGO log levels

Log on as root or egoadmin.

If you need to troubleshoot or debug your system, you need to change log levels from the default LOG_WARNING to a log level that gives more details (we suggest LOG_DEBUG). You can also enable or disable logging completely.

Separate log levels are set for each daemon. To change the log level for an entire cluster, you must set each daemon individually.

Logging level settings are not permanent and do not change `ego.conf`. When the cluster shuts down or restarts, logging level settings do not persist. Instead, log levels reset to LOG_WARNING.

Refer to the egosh command for details on all the command options.

Note:

If you are setting the log level for VEMKD and this daemon fails over, the permanent setting in ego.conf (Warning level) becomes the current setting. To change it, follow these steps once more.

1. To enable the more detailed log level LOG_DEBUG, run `egosh debug` *daemon* `on`.

   For example, `egosh debug vemkdon`.

   You can also specify a log class using the -c option if you want to receive entries solely concerning one log class (like authentication or memory). Refer to the `egosh` command for details on using this option.

   You have enabled log level LOG_DEBUG for vemkd. All entries are logged in `vemkd.log.`*`hostname`*.
2. To return the log level to the default LOG_WARNING, run `egosh debug` *daemon*`off`.

   For example, `egosh debug vemkdoff`.

   You have returned your log level to LOG_WARNING (the default) for vemkd. All entries are logged in `vemkd.log.`*`hostname`*.

You should debug your system as quickly as possible and set your log level to warning for daily activity.

Caution:

The LOG_DEBUG logging level is very verbose and fills up your log files quickly, which can adversely affect performance.

Change EGO log file locations

On Windows machines, you can save log files to a specified location. If you are running a clustered application manager (such as Platform LSF), you want to ensure the location is the same for both applications (both LSF and EGO).

1. Open `%EGO_CONFDIR%\kernel\conf\ego.conf` (or `%EGO_CONFDIR%\ego.conf`).
2. To enable logging in Windows, you must define the parameter EGO_LOGDIR=*dir*, where *dir* specifies the EGO system log file directory.

   Error messages from all servers are logged into files in this directory. To effectively use debugging, set EGO_LOGDIR to a directory such as `C:\Temp`.
3. To enable logging of error messages into the directory files specified by EGO_LOGDIR, you must also define the related parameter EGO_LOGDIR_USE_WIN_REG=**n**.

   If you do not define EGO_LOGDIR_USE_WIN_REG, or if you define it with a value other than **n**, then EGO logs error messages to the default local directory specified in the Windows registry key:

   HKEY_LOCAL_MACHINE\SOFTWARE\Platform Computing Corporation\EGO \EGO_LOGDIR

   Note:

   For 64-bit Windows machines, EGO logs error messages to the default local directory specified in this Windows registry key: HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node \Platform Computing Corporation\EGO\EGO_LOGDIR.

   If a server is unable to write in the EGO system log file directory, EGO attempts to write to these directories:

   - Linux, in this order:
     1. $TMPDIR (if variable is defined)
     2. /tmp
   - Windows, in this order:
     1. OS default temp directory, by call Windows API GetTempPath()
     2. System directory (C:\)

Change service director (named) log levels

Log on as root or egoadmin.

Log levels for the service director are set differently than log levels for the service controller, Platform Management Console, and web service gateway. Logged by the BIND DNS server, the service director logs information gathered during the updating and querying of service instance location.

The default log level for the service director is set to severity debug 3. If your system is running well, you likely want to set the severity to something lower. Only use severity debug [level], which generates a lot of messages, for troubleshooting or debugging purposes. In this case, you may wish to reduce the associated log class as well.

1. To reduce the amount of detail logged to `named.log`, open the file from this location:
   - Windows: `EGO_TOP\eservice\esd\conf\named\conf\named.conf`
   - Linux: `EGO_TOP/eservice/esd/conf/named/conf/named.conf`
2. Locate the section on logging, and the subsection on severity.

```
};logging {       channel my_file {             file "named.log" versions
2 size 10m;               severity debug 3;              print-time
yes;             print-category yes;                  };       category
default {my_file;};};
```

3. Change the severity log class to a lower class.

   Choose from critical, error, warning, notice, info, debug, or dynamic (where critical logs only the most critical event information, resulting in the least number of log entries).

   If you choose to keep the default log class (debug), you can lower the log level from 3 to 1 (where 1 results in the fewest messages and the lowest detail level of logged debug information).
4. Save the file.

Enable EGO event logging for auditing purposes

You must be a cluster administrator to perform this task. You have previously run `egoconfig mghost` *shared_dir* during the installation for multi-host clusters or UNIX installations.

EGO monitors and logs security-sensitive events related to EGO services and to host, user, and consumer containers. By default, auditing of these events is disabled. To collect information to better monitor system security, enable logging by configuring the `ego.conf`, `egosc.conf.xml`, and `rs.xml` files.

Note the following:

- Audit logs can be enabled independently of each other.
- For multi-host clusters, configure files from within the shared directory, not the local directory (local directory configurations are ignored). For single-host Windows clusters, you can configure local files.

Important:

UNIX installations require a shared directory.

- Only master hosts perform audit logging; compute hosts do not normally have access to the shared locations where configuration files are stored. You never need to enable audit logging (configure files) on compute hosts.

1. To enable logging for auditing of core EGO functions (for example, security):
   a) Open `ego.conf`.
      - On Windows: `$EGO_CONFDIR\ego.conf`
      - On Linux: `$EGO_CONFDIR/ego.conf`

   b) Turn on EGO audit logging by adding the following parameter:

      EGO_AUDIT_LOG=**Y**

   c) At this time, you may also want to define an audit log directory by configuring the EGO_AUDIT_LOGDIR parameter. This is the default directory location and name:
      - On Windows: `EGO_AUDIT_LOGDIR=$EGO_CONFDIR\audits`
      - On Linux: `EGO_AUDIT_LOGDIR=$EGO_CONFDIR/audits`

      Note:

      You can change the name, but the location must be a shared directory; ensure there are no spaces in the directory name.

      Once defined, the vemkd and egosc daemons automatically create the directory.

   d) Save and close the file.

      Note that there is no automatic file roll-over or audit log cleanup. Ensure that you manually manage the file size.
2. To enable audit logging for the service controller:
   a) Open `egosc.conf.xml`
      - On Windows: `EGO_TOP\\eservice\esc\conf\egosc.conf.xml`
      - On Linux: `EGO_TOP//eservice/esc/conf/egosc.conf.xml`

   b) Turn on the EGO service controller log (`egoservice.audit.log`) by adding the following element:

<ESC_AUDIT_LOG>**ON**</ESC_AUDIT_LOG>

c) Save and close the file.

3. To enable audit logging for the repository service:

   a) Open `rs.xml`.

      - On Windows: `EGO_TOP\eservice\esc\conf\services\rs.xml`
      - On Linux: `EGO_TOP/eservice/esc/conf/services/rs.xml`

   b) Turn on the repository service audit log by adding the following element:

```
<ego:EnvironmentVariable name="RS_AUDIT_LOG">ON</
ego:EnvironmentVariable
>
```

      Note:

      The default setting is OFF. The setting is case sensitive.

      The RS logs information into the configured audit log directory, as specified by the parameter EGO_AUDIT_LOGDIR defined in `ego.conf`. If this parameter is not found or defined, the RS logs to this directory:

      - On Windows: `EGO_TOP\audits`
      - On Linux: `EGO_TOP/audits`

   c) Save and close the file.

   d) Stop the RS service.

      egosh service stop RS

4. Restart EGO on the master host.

   egosh ego restart

   EGO restarts any currently stopped services. Changes made to stopped services now take effect.

View EGO log files

Event and audit log files can be viewed through the Platform Management Console.

1. From the Platform Management Console, navigate to System Logs.
2. Select Standard Logs.
3. From Logs (List), select a log from the list (for example, “lim.log”).

   Logs of the selected category are listed for every management host in the cluster.
4. Under Log retrieval parameters, specify the following:
   a) Optional. Filter the available hosts by typing full or partial host names in the Filter available hosts field.

      The list of available hosts is dynamically filtered as you type to show those hosts containing the same characters/name.

   b) Choose which hosts you want to retrieve and view this log from (for example, all hosts for which you want to view “lim.log” from).

      1. Click a host name from the Available hosts list.
      2. Click Add - > to move this host to the Retrieve logs from these hosts list.
      3. Do this for as many hosts as you want to retrieve logs from.

      Note:

      For performance reasons, there is a limit to the number of hosts you can add to the list. A message alerts you when this limit is reached.
5. Click Retrieve Log List.
6. Specify how much of the retrieved log you want to see. Options for retrieved data volume include the following:

   - Complete log
   - Last number of lines (a specified number of lines in the file); if you choose to retrieve a specified number of lines, enter the number in the field beside the drop-down list

   Note:

   You only see the last number of specified lines in the file, not the first.

   A list of logs from the requested hosts displays, along with details including the host on which the log file is located, the log file size, and the date of the last entry in the log file.
7. Click a specific log file to view or save.

   Choose to open and view the log file in a specified program, or save it to disk.

   Note:

   Log files cannot be edited, renamed, or deleted from within the Platform Management Console.

EGO log file properties

Log properties in the Console include the following (as seen on the Logs (List) page):

Property	Definition
Category name	Identifies the log name.
Summary/Description	Provides a brief summary of the log purpose and the information that is written to the log file.
System Component	Identifies from what product system component this log originates from. For example, the lim logs originate in the EGO component.

Troubleshoot using multiple log files

Log file locations and content

If a service does not start as expected, open the appropriate service log file and review the run-time information contained within it to discover the problem. Look for relevant entries such as insufficient disk space, lack of memory, or network problems that result in unavailable hosts.

Log file	Default location	What it contains
catalina.out	Linux: *`EGO_TOP`*`/gui/tomcat/logs/catalina.out` Windows: *`EGO_TOP`*`\gui\tomcat\logs\catalina.out`	Logs system errors and debug information from Tomcat web server startup.
ego.audit.log	Linux:`EGO_AUDIT_LOGDIR=` *`EGO_TOP`*`/audits/`ego.audit.log Windows: `EGO_AUDIT_LOGDIR=` *`EGO_TOP`*`\audits\ego.audit.log`	Logs all events related to the host, consumer, and user, including opening and closing a host, adding, modifying, deleting a consumer or resource plan, user role changes and user logging actions. Must be enabled first.
egoservice.audit.log	Linux: `EGO_AUDIT_LOGDIR=` *`EGO_TOP`*`/audits/`egoservice.audit.log Windows: `EGO_AUDIT_LOGDIR=` *`EGO_TOP`*`\audits\egoservice.audit.log`	Logs all events related to the starting and stopping of all services. Must be enabled first.
esc.log	Linux: *`EGO_TOP`*`/eservice/esc/log/esc.log.`*`hostname`* Windows: *`EGO_TOP`*`\eservice\esc\log\esc.log.`*`hostname`*`.txt`	Logs service failures and service instance restarts based on availability plans. Errors surrounding Platform Management Console startup are logged here.
lim.log	Linux: *`EGO_TOP`*`/kernel/log/lim.log.`*`hostname`* Windows: *`EGO_TOP`*`\kernel\log\lim.log.`*`hostname`*`.txt`	Logs static and dynamic attribute information about individual hosts. Logs any error warnings generated while lim was running. If log class set to debug, logs information that helps to locate the encountered problem.
named.log	Linux: • *`EGO_TOP`*`/eservice/esd/conf/named/namedb/named.log`, or • *`EGOshare`*`/eservice/esd/conf/named/namedb/named.log` Windows: • *`EGO_TOP`*`\eservice\esd\conf\named\namedb\named.log`, or • *`EGOshare`*`\eservice\esd\conf\named\namedb\named.log`	Logs information gathered during the updating and querying of service instance location; logged by BIND, a DNS server.

Log file	Default location	What it contains
pem.log	Linux: *`EGO_TOP`*`/kernel/log/pem.log.`*`hostname`* Windows: *`EGO_TOP`*`\kernel\log\pem.log.`*`hostname`*`.txt`	Logs remote operations (start, stop, control activities, failures). Logs tracked results for resource utilization of all processes associated with the host, and information for accounting or chargeback.
pim.log	Linux: *`EGO_TOP`*`/kernel/log/pim.log.`*`hostname`* There are no pim log files for Windows.	Logs resource usage information about processes running on the host.
vemkd.log	Linux: *`EGO_TOP`*`/kernel/log/vemkd.log` Windows: *`EGO_TOP`*`\kernel\log\vemkd.log.txt`	Logs aggregated host information about the state of individual resources, status of allocation requests, consumer hierarchy, resources assignment to consumers, and started operating system-level process.
wsg.log	Linux: • *`EGO_TOP`*`/eservice/wsg/log/wsg.log`, or • *`EGOshare`*`/eservice/wsg/log/wsg.log` Windows: • *`EGO_TOP`*`\eservice\wsg\log\wsg.log`, or • *`EGOshare`*`\eservice\wsg\log\wsg.log`	Logs service failures surrounding web services interfaces for web service clients (applications).
wsm.log	Linux: *`EGO_TOP`*`/gui/logs/wsm.log.`*`hostname`* Windows: *`EGO_TOP`*`\gui\logs\wsm.log.`*`hostname`*`.txt`	Logs information collected by the web server monitor daemon. Failures of the WEBGUI service that runs the Platform Management Console are logged here.

Matching service error messages and corresponding log files

If you receive this message while using EGO…	This may be the problem…	Review this log file
`failed to create vem working directory`	Cannot create work directory during startup	vemkd
`failed to open lock file`	Cannot get lock file during startup	vemkd
`failed to open host event file`	Cannot recover during startup because cannot open event file	vemkd
`lim port is not defined`	EGO_LIM_PORT in `ego.conf` is not defined	lim
`master candidate can not set GET_CONF=lim`	Wrong parameter defined for master candidate host (for example, EGO_GET_CONF=LIM)	lim

If you receive this message while using EGO…	This may be the problem…	Review this log file
`there is no valid host in EGO_MASTER_LIST`	No valid host in master list	lim
`ls_getmyhostname fails`	Cannot get local host name during startup	pem
`/tmp directory (%s) not exist or not accessible, exit`	/tmp directory does not exist	pem
`incorrect EGO_PEM_PORT value %s, exit`	EGO_PEM_PORT is a negative number	pem
`chdir(%s) fails`	/tmp directory does not exist	esc
`cannot initialize the listening TCP port %d`	Socket error	esc
`cannot log on`	Log on to vemkd failed	esc
`JAVA_HOME is not defined, exit`	WEBGUI service profile is wrong	wsm
`failed to get hostname: %s`	Host name configuration problem	wsm
`event_init ( ) failed`	EGO event plugin configuration problem in `ego.conf` file	wsm
`ego.conf_loadeventplug ( ) failed`	Event library problem	wsm
`cannot write to child`	Web server is down or there is no response	wsm
`child no reply`	Web server is down or there is no response	wsm
`vem_register: error in invoking vem_register function`	VEM service registration failed	wsg
`you are not authorized to unregister a service`	Either you are not authorized to unregister a service, or there is no registered client	wsg
`request has invalid signature: TSIG service.ego: tsig verify failure (BADTIME)`	Resource record updating failed	named

Traces

Traces enable an administrator to turn logging on for information specific to an object in EGO. Currently, you can only set traces for activities. An activity is a hosting environment for a service; it provides the context for a service.

Setting a trace results in log entries for that object at log level LOG_TRACE.

Set a trace

Log on as root or egoadmin.

If you need to troubleshoot a specific object in EGO, you can set a trace for it. Setting a trace logs entries for that object with log level LOG_TRACE.

1. Run `egosh debug` *`daemon`*`on` *`host`* `-t -c` *`LOG_CLASS`* `-o "`*`key=value`*`"`.

   ```
   egosh debug pemon hostA -t -c LC_CLIENT -o "activity=114"
   ```

   Specifying the logging class is optional.
2. To turn off the trace, run `egosh debug` *daemon*off *host* -t -c *LOG_CLASS* -o "*key=value*".

   ```
   egosh debug pemoff hostA -t -c LC_CLIENT -o "activity=114"
   ```

   Once you turn off the trace, the log level resets to its default.

CHAPTER

18

Events

EGO events

EGO events can be monitored and used to trigger actions automatically.

EGO events belong to the following categories:

- System events, which identify occurrences within the cluster
- Platform Management Console events, which identify occurrences that affect the web server or the Platform Management Console itself.

System events

Event name	Default level	Triggered when …
SYS_CLS_UNLICENSED • Component name: ego_lim • Returned integer: 0	Error	The cluster is not licensed
SYS_HOST_CLOSED • Component name: ego_vemkd • Returned integer: 6	Warning	A host is closed
SYS_HOST_UNAVAIL • Component name: ego_vemkd • Returned integer: 4	Warning	A host becomes unavailable
SYS_PEM_DOWN • Component name: ego_lim • Returned integer: 2	Error	Local pem goes down
SYS_PEM_UP • Component name: ego_pem • Returned integer: 12	Info	Local pem is started
SYS_SVC_DOWN • Component name: ego_sc • Returned integer: 7	Error	A system service goes down
SYS_SVC_UP • Component name: ego_sc • Returned integer: 8	Info	A system service is started

Event name	Default level	Triggered when …
SYS_SVC_INST_DOWN - Component name: ego_sc - Returned integer: 9	Error	An instance of a system service goes down
SYS_SVC_INST_UP - Component name: ego_sc - Returned integer: 10	Info	An instance of a service is started
SYS_VEMKD_DOWN - Component name: ego_lim - Returned integer: 1	Error	vemkd goes down
SYS_VEMKD_UP - Component name: ego_vemkd - Returned integer: 11	Info	vemkd is started

Platform Management Console events

Event name	Default level	Triggered when …
SYS_GUI_CPU_HI_WATER_MARK - Component name: GUI - Returned integer: 3	Warning	The web server host utilization exceeds the threshold set for `CPU_HIGH_MARK` in `wsm.conf`
SYS_GUI_MEMORY_HI_WATER_MARK - Component name: GUI - Returned integer: 2	Warning	The web server memory usage exceeds the threshold set for `MEM_HIGH_MARK` in `wsm.conf`

Enable events

Log on as root or egoadmin

If you want to be notified about cluster events, enable this feature. By default, events are not enabled.

1. In `ego.conf`, remove the comment ("#") from the parameters EGO_EVENT_MASK and EGO_EVENT_PLUGIN.

   You have enabled events to be logged.
2. Set EGO_EVENT_MASK to the log level you want.

   EGO_EVENT_MASK must be set to one of the following values:

   - LOG_ERR: Provides information about error events only
   - LOG_WARNING: Provides information about warning and error events
   - LOG_INFO: (Default) Provides information about all events

   For example, **EGO_EVENT_MASK=LOG_INFO**.
3. Set EGO_EVENT_PLUGIN to the plug-in configuration you want to use.

   You can use the provided SNMP configuration or specify your own.

   a) You can modify the default port (port 162) by specifying `TRAPPORT=`*`port_number`* in the string.

      ```
      EGO_EVENT_PLUG_IN="eventplugin_snmp[…,TRAPPORT=port_number]"
      ```

      Note that in a Linux environment, do not use quotation marks around the event plug-in definition.

      For example, EGO_EVENT_PLUG_IN=eventplugin_snmp […,TRAPPORT=*port_number*].
4. Save `ego.conf` and restart your cluster.

You should specify your SNMP plug-in and create a trap.

View an event

You must enable events to be logged before you complete this task. Log on as root or egoadmin.

To view an event, you need to set your SNMP plug-in and trap information.

An SNMP plug-in is installed but not enabled by default. You can replace this SNMP plug-in with a similar plug-in. We recommend SNMP Trap Watcher by BTT Software.

1. Open `ego.conf` for editing.

   You need to specify two parameters under the EGO event configuration section: EGO_EVENT_PLUGIN and EGO_EVENT_MASK.
2. Enable events to be logged.

   In the `EGO event configuration` section, remove the comment ("#") from the parameter EGO_EVENT_PLUGIN and specify the plug-in information.

   There is a plug-in installed and specified by default.
3. (Optional) Change the default port (162) by specifying `TRAPPORT=`*`port_number`* in your event plug-in string.
4. Set up an SNMP v1 trap for EGO events by specifying the name and configuration file location with your SNMP information.

   For example: `EGO_EVENT_PLUGIN=eventplugin_snmp[SINK=,MIBDIRS=/mibs]` *host EGO_CONFDIR*

   where *host* represents the name of the host where the SNMP trap daemon is running.

   Note the following:

   - The MIBDIRS directory may also equal `MIBDIRS=$EGO_CONFDIR/kernel/conf/mibs`.
   - In a Windows environment, use quotation marks around the event plug-in definition.

     For example, `EGO_EVENT_PLUGIN="eventplugin_snmp[SINK=`*`host`*`,MIBDIRS=$EGO_CONFDIR\kernel\conf\mibs]"`.
5. Set EGO_EVENT_MASK=LOG_INFO.
6. Save `ego.conf` and restart your cluster.

You need to configure your SNMP manager to take an action for events that are received.

Responding to event message sys_cls_unlicensed

This is a system event triggered when the cluster is not licensed. By default, this event is set to an Error level.

1. Ensure that the EGO_LICENSE_FILE variable in `ego.conf` is porting to the correct file location on the master host.
2. Review the license file content and ensure it is valid. Confirm that the license has not expired.

PART

III

Scenarios and Tutorials

Tutorial: Creating initial resource groups

Goal

Setting up useful resource groups is essential to making full and efficient use of cluster capabilities. Following the steps below, you create resource groups that set aside specific hosts for management duties and divvy up the remainder of your hosts based on maximum memory.

Description

You have added most of your hosts to your cluster but have not yet set up an extensive resource plan, created new resource groups, or modified default resource groups. You are preparing to customize the plan for your applications and want to divide your hosts by memory as you expect to run varied workload units with some requiring not less than 1000 MB of maximum memory and others requiring very little memory at all. You want to ensure the following for your workload units:

- They have access to hosts with the necessary amount of maximum memory
- They have no need to wait for appropriate hosts to become available
- Those workload units that require very little memory do not get hosts with a large max mem

At a glance

1. Plan your groups
2. Check the ManagementHosts resource group
3. Review and modify the master host candidate list
4. Create new dynamic resource groups
5. Modify your resource plan for new resource groups
6. How to grow: Advanced resource groups

Plan your groups

Resource groups overview

Resource groups are logical groups of hosts. Resource groups provide a simple way of organizing and grouping resources (hosts) for convenience; instead of creating policies for individual resources, you can create and apply them to an entire group. Groups can be made of resources that satisfy a specific static requirement in terms of OS, memory, swap space, CPU factor, and so on, or that are explicitly listed by name.

The cluster administrator can define multiple resource groups, assign them to consumers, and configure a distinct resource plan for each group. For example:

- Define multiple resource groups: A major benefit in defining resource groups is the flexibility to group your resources based on attributes that you specify. For example, if you run workload units or use applications that need a Linux OS with not less than 1000 MB of maximum memory, then you can create a resource group that only includes resources meeting those requirements.

Note:

No hosts should overlap between resource groups. Resource groups are used to plan resource distribution in your resource plan. Having overlaps causes the hosts to be double-counted (or more) in the resource plan, resulting in recurring under-allocation of some consumers.

- Configure a resource plan based on individual resource groups: Tailoring the resource plan for each resource group requires you to complete several steps. These include adding the resource group to each desired top-level consumer (thereby making the resource group available for other sub-consumers within the branch), along with configuring ownership, enabling lending/borrowing, specifying share limits and share ratio, and assigning a consumer rank within the resource plan.

Resource groups are either specified by host name or by resource requirement using the select string.

By default, EGO comes configured with three resource groups: InternalResourceGroup, ManagementHosts, and ComputeHosts. InternalResourceGroup and ManagementHosts should be left untouched, but ComputeHosts can be kept, modified, or deleted as required.

Note:

Requirements for resource-aware allocation policies (where only certain resources that meet specified requirements are allocated to a consumer) can be met by grouping resources with common features and configuring them as special resource groups with their own resource plans.

Gather the facts

You need to know which hosts you have reserved as management hosts. You identified these hosts as part of the installation and configuration process. If you want to select different management hosts than the ones you originally chose, you must uninstall and then reinstall EGO on the compute hosts that you now want to designate as management hosts (a master host requires installing the full package), and then run `egoconfig mghost`. The tag mg is assigned to the new management host, in order to differentiate it from a compute host. The hosts you identify as management hosts are subsequently added to the `ManagementHosts` resource group.

Management hosts run the essential services that control and maintain your cluster and you therefore need powerful, stable computers that you can dedicate to management duties. Note that management hosts are expected to run only services, not to execute workload units.

Ensure that you designate one of your managements host as the master host, and another one or two hosts as failover candidates to the master (the number of failover candidates is up to you, and may depend on the size of your production cluster).

1. Make a list of hosts that have been installed with the full package, and that have the tag mg assigned to them (from having run `egoconfig mghost`).

   You should be able to get a list from the person who installed your cluster.
2. Review the list of management hosts.

   Ask yourself if these are your most trusted hosts with the reliability they need to be responsible for the entire cluster.
3. (Optional) Remove any listed management hosts you do not trust.
   a) If you have configured automatic startup during your cluster setup, then run `egoremoverc.sh`.

   Doing this prevents automatic startup when the host reboots, which keeps the host from being re-added dynamically to the cluster.

   b) Run `egoconfig unsetmghost` to remove the host from the management host group.

      Running this command removes the host entry from ego.cluster.*cluster_name*.

   c) If the host is a master candidate, run `egoconfig masterlist` to remove the host from the failover order.

   d) Restart the master host to change the local host from a management host to a compute host, and for the cluster file to get read again.

4. (Optional) Designate different management hosts.

   a) For each Linux host you wish to designate as a management host, including master candidates, do the following:

      1. Run the `egoconfig mghost` command:

         egoconfig mghost *EGOshare*

         where *EGOshare* is the shared directory that contains important files such as configuration files to support master host failover (once the `egoconfig mghost` command is run and the files are copied over).

         For example, `egoconfig mghost /share/ego`

         Note that the shared directory is the same for all management hosts.

      2. Set the environment on the local host so that *`EGO_CONFDIR`* gets set properly and the changes take effect.

         Doing this changes *`EGO_CONFDIR`* from a local to shared directory.

      3. Restart the master host so that the cluster file gets read again.

   b) For each Windows host you wish to designate as a management host, including the master candidates, do the following:

      1. Run the `egoconfig mghost` command:

         egoconfig mghost *EGOshare domain_name\user_name password*

         where *EGOshare* is the shared directory that contains important files such as configuration files to support master host failover (once the `egoconfig mghost` command is run and the files are copied over), *user_name* is the egoadmin account, and *password* is the egoadmin password.

         For example, `egoconfig mghost \\Hostx.mycompany.com\EGO\share mycompany.com\egoadmin mypasswd`

         Note:

         The shared directory is the same for all management hosts. Also, be sure to use a fully qualified domain name.

      2. Restart the master host so that the cluster file gets read again.

You now have a list of hosts you would like as management hosts. You use this list to check hosts that actually belong to the management hosts resource group.

Check the ManagementHosts resource group

You must be logged on to the Platform Management Console as a cluster administrator.

The `ManagementHosts` resource group is created during the installation and configuration process. Each time you install and configure the full package on a host, that host is statically added to the `ManagementHosts` resource group.

You need to ensure that the trusted hosts you identified in the section Gather the facts (above) are the same as the hosts that were configured to be management hosts.

1. From the Platform Management Console, click Resources > Configure Resource Groups > Resource Groups.

   A list of all resource groups displays.

   By default, your resource groups are `ComputeHosts`, `InternalResourceGroup`, and `ManagementHosts`.
2. From the list, click ManagementHosts.

   The properties for `ManagementHosts` display.

   Caution:

   Do not, under any circumstances, modify any of the `ManagementHosts` properties (except for the description). You could seriously damage your cluster.
3. Note and compare the hosts listed in the Member hosts section at the bottom.

   The hosts that are members of the `ManagementHosts` resource group are listed here.

   Do these hosts match the list of hosts you made in the section Gather the facts? If not, contact the person in charge of installation and make sure each management host is configured properly.

   You need the exact host name(s) for the next topic.

You have made sure the hosts you want as management hosts belong to the `ManagementHosts` resource group. The installation and configuration matches your desired cluster setup.

Review and modify the master host candidate list

You must be logged on to the Platform Management Console as a cluster administrator.

Once you have reviewed your `ManagementHosts` resource group, you need to make sure your master host candidate list is correct.

1. Select Cluster > Summary.

   A summary displays.
2. Click Master Candidates.

   The master host is the first host in the list displayed in the right column. Other host names may be listed as candidates or as available hosts (right and left columns, respectively).
3. Review master and candidates.

   The master host is the host listed first in the candidates column. All others under the candidate list should be eligible hosts that are also part of the `ManagementHosts` resource group.

   a) Check the host names against the list you made when you checked the `ManagementHosts` resource group.

   b) Use the controls to move hosts around. Add any hosts that you want as master candidates into the candidates column in the order you want them to failover.

      You cannot remove the master host.

Create new dynamic resource groups

You must be logged on to the Platform Management Console as a cluster administrator. You should not be running any workload units while you perform this task; when you delete a resource group, those hosts are no longer assigned to a consumer. You should complete this task before changing your resource plan for the first time. If you have modified the resource plan and want to save those changes, export the resource plan before starting this task.

You can create resource groups that automatically place all your compute hosts in two (or more) different resource groups. You can split your hosts up this way if some of the applications or workload units you plan to run on the EGO cluster have distinct or important memory requirements.

You can logically group hosts into resource groups based on any criteria that you find important to the applications and workload units intend to run. For example, you may wish to distinguish hosts based on OS type or CPU factor.

1. Select Resources > Configure Resource Groups > Resource Groups.

   A list of your existing resource groups displays.

   By default, your resource groups are `ComputeHosts`, `InternalResourceGroup`, and `ManagementHosts`.
2. Delete the resource group called `ComputeHosts`.

   To do this, select Actions > Delete Resource Group from the `ComputeHosts` row. When asked, confirm the deletion.

   Caution:

   The `InternalResourceGroup` and `ManagementHosts` groups should never be deleted. They are special resource groups that contains hosts used for EGO services.
3. Select Global Actions > Create a Resource Group.

   The resource group properties display.
4. Fill in the resource group properties.
   a) Type a name that describes the hosts that you are going to select for this group. In this example, we use “maxmem_high”.
   b) Select 1 slot per CPU.
   c) Make sure the resource selection method is Dynamic (Requirements).
   d) Under Hosts to Show in List, select Hosts filtered by resource requirement.
   e) In the Resource Requirement String field, type **select(!mg && maxmem > 1000)**.

      The command select ignores any hosts belonging to the `ManagementHosts` resource group (**!mg**) and add any non-management host that has a maximum memory of 1001 MB or more (**maxmem > 1000**).
   f) Click Refresh Host List.

      In the Member hosts section, a list of any hosts (as found in the current cluster) that meet the requirements you specified with the select string is generated.

   g) Review the hosts in the member section and make any modifications you need to the select string until the member list is correct.

      Only hosts that currently match the requirements are displayed here. However, the list is dynamic. As you add hosts to the cluster that meet these requirements, they are automatically added to this resource group.
   h) Click Check for overlaps in the member hosts section to make sure the member hosts do not belong to any other resource groups.

      If you have overlaps, modify your selection string until overlaps no longer exist. Hosts must never overlap between resource groups. Having overlaps causes the hosts to be double-counted (or more) in the resource plan, resulting in recurring under-allocation of some consumers. The exception is with hosts listed in `InternalResourceGroup`—although all hosts in the cluster are listed here, they are not "double-counted" in the resource plan.
   i) Once you have no overlaps, click Create.
5. Click Resource Groups again.

   A list of resource groups displays, including the maxmem_high group you just created.
6. Create a second resource group.

   Follow the same steps above with the following differences.
   a) Name the second resource group "maxmem_low".
   b) Add the selection string **select(!mg && !(maxmem > 1000))**.

      This resource group is now made up of any compute host not belonging to the `ManagementHosts` resource group and excluding hosts you specified for the maxmem_high resource group.

      We recommend that you specify one resource group that excludes all other resource groups or selection string requirements (specify using "not" (!)). That way, all your hosts fall into one resource group or another.

   You have now deleted the `ComputeHosts` resource group and split all your hosts, except those belonging to the `ManagementHosts` resource group, into two new groups: one made up of hosts with memory over 1000 MB (maxmem_high) and one made up of all other hosts with memory of 1000 MB or less (maxmem_low).

Modify your resource plan for new resource groups

If you know you intend to create more resource groups, do that first even if you do not know all the details of the resource groups.

Any time you add, modify, or delete a resource group, you need to manage resource distribution for these resource groups using the resource plan.

1. Click Consumers > Consumers & Plans > Resource Plan.
2. Use the Resource Group drop-down menu to switch between resource groups and modify your resource plan details for each resource group.

Note:

Resources groups that do not yet have consumers assigned to them do not appear in the drop-down menu. Consumers must first be assigned from the Consumers & Plans > Consumers page.

Never make any changes to the `ManagementHosts` resource group in the resource plan.

How to grow: Advanced resource groups

Now that you have basic resource groups (one for your management hosts and two for your compute hosts) you can begin to specialize and split up one resource group that is based on available memory.

For example, if you know that an application you run requires not only machines with 1001 MB of available memory or more, but also two or more CPUs, you can create a new resource group (and then modify the existing “maxmem_high” resource group) to make these specific resources available to any consumer. The new resource group “maxmemhighmultiCPU” would have the selection string:

select(!mg && maxmem > 1000 && ncpus>=2)

You would then modify the existing resource group “maxmem_high” to read:

select(!mg && !(ncpus>=2) && maxmem > 1000)

As a result, the maxmem_high group uses only single CPU hosts.

Tutorial: Building Your Tree

Goal

Setting up a basic consumer tree that mirrors your business structure vastly increases your ability to manage your resources efficiently. By following the steps below, you gain an understanding of a consumer tree, consumers, and how to plan for the future.

Description

You have recently installed your cluster and have not yet created consumers, modified the resource plan, or modified, removed, or added resource groups. You want to understand what your consumer tree does and how you should build it.

At a glance

1. Gather the facts
2. How your business structure maps to your tree
3. Consumer tree
4. Create your consumers
5. How you are prepared

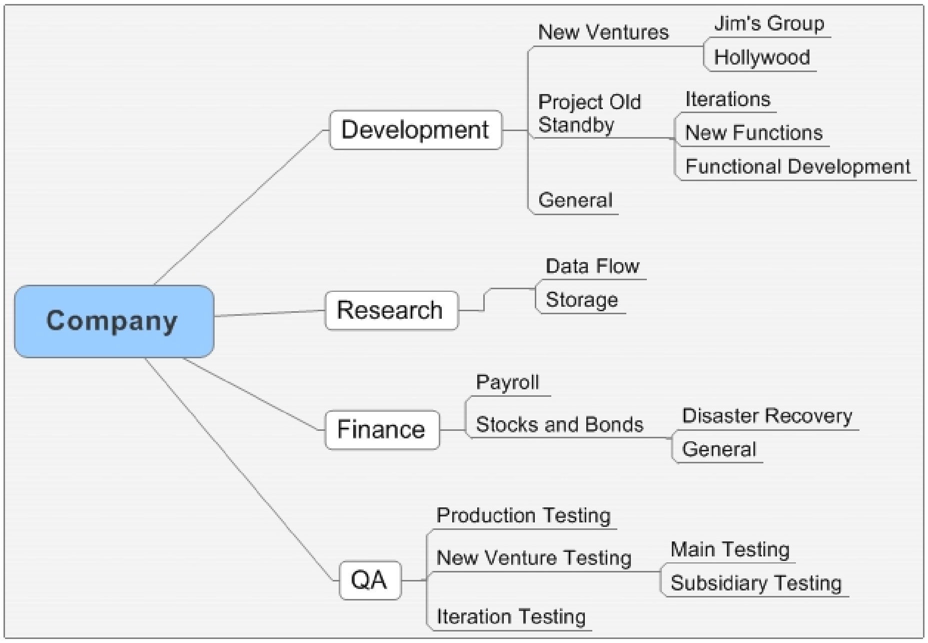
Gather the facts

Before you begin to create your consumers and build your tree, you need to know how you want to control your cluster resources. To begin the process, you need to map out your current business structure.

1. Map out your business structure by hand.

   As a best practice, restrict the number of levels to four.

   For example, your business structure might look like this:



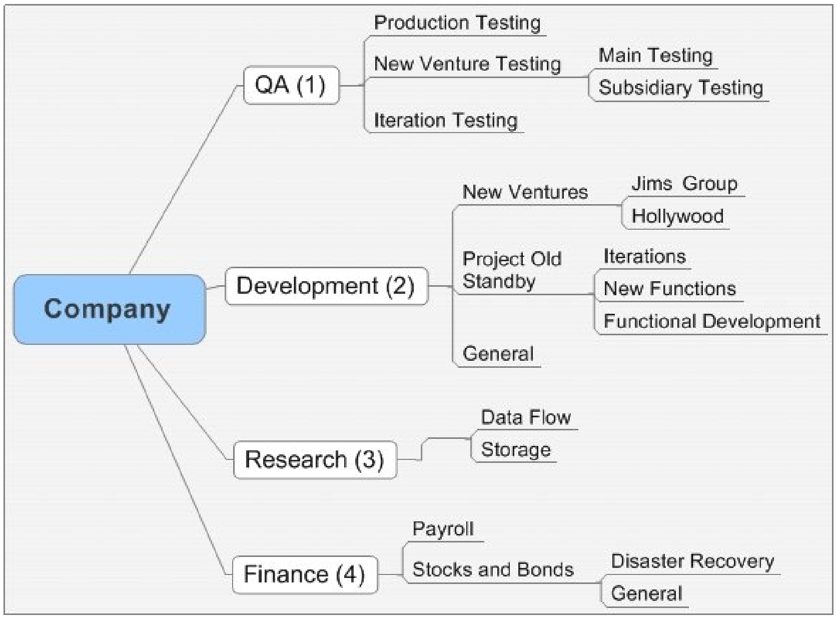
2. Prune your tree.

   Once you have a detailed diagram of your company structure, decide if any branches of your tree do not need to consume cluster resources, and remove them from your diagram.

3. Prioritize your business processes.

   a) For each top-level business process, decide which areas should receive resources first when they need them.

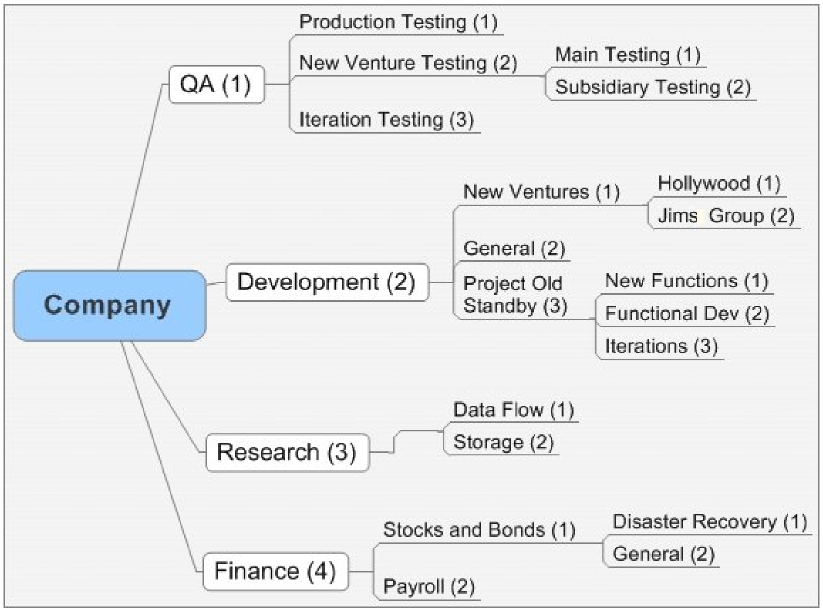
      If I decide that QA should have priority over everything else, and Finance likely needs resources with less urgency, I might set the order like this:



      Keep in mind that you want to make sure that your business structure makes a distinction that parallels how you want to manage and distribute your resources. You may want to break out special projects that need dedicated or specialized hosts.

      Your business structure and its hierarchy should reflect long-term business goals because it can be complicated to modify the tree later on.

   b) Prioritize all your lower level business areas relative to other leaf consumers from the same branch (siblings).



You now have the basic planning information that you are going to use to create your consumer tree.

How your business structure maps to your tree

The business structure you mapped out above becomes the template to create a consumer tree.

The top-level business processes become top-level consumers.

The lowest area of business becomes a leaf consumer and this is the consumer location where you register such things as services or other application managers.

You have some built-in consumers that run necessary cluster services (ManagementServices and Cluster services, and their respective leaf consumers, EGOManagementServices and EGOClusterServices).

You also have a SampleApplications consumer and its leaf consumer, EclipseSamples, used by the Sample client applications included in the EGO SDK.

About the consumer tree

The consumer tree organizes consumers into a structure that makes it easy to apply resource plans.

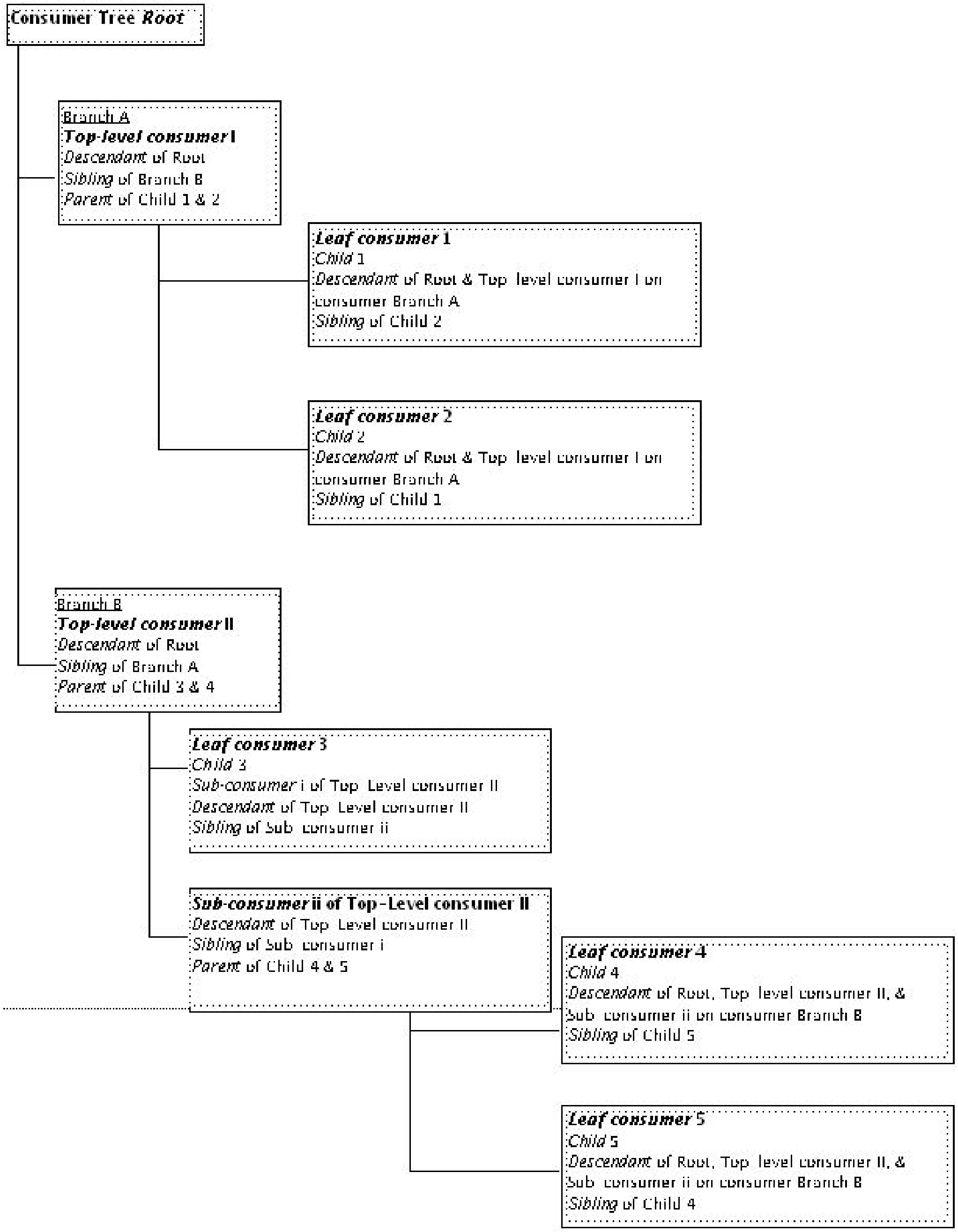
Overview

The consumer tree is closely related to the resource plan. The plan cannot be defined without the tree.

The tree only defines organizational relationships among consumers, while the plan defines resource allocation.

The choice of consumers and their hierarchy should reflect long-term business goals because it can be complicated to modify the tree. To make the system adjust to short-term business changes, you can modify the users associated with a consumer, or the resource plans defined in the plan.

Graphical description


Consumer Tree Root
Branch A
Top-level consumer I
Descendant of Root
Sibling of Branch B
Parent of Child 1 & 2
Leaf consumer 1
Child 1
Descendant of Root & Top-level consumer I on consumer Branch A
Sibling of Child 2
Leaf consumer 2
Child 2
Descendant of Root & Top-level consumer I on consumer Branch A
Sibling of Child 1
Branch B
Top-level consumer II
Descendant of Root
Sibling of Branch A
Parent of Child 3 & 4
Leaf consumer 3
Child 3
Sub-consumer i of Top-Level consumer II
Descendant of Top-Level consumer II
Sibling of Sub-consumer ii
Sub-consumer ii of Top-Level consumer II
Descendant of Top-Level consumer II
Sibling of Sub-consumer i
Parent of Child 4 & 5
Leaf consumer 4
Child 4
Descendant of Root, Top-level consumer II, & Sub-consumer ii on consumer Branch B
Sibling of Child 5
Leaf consumer 5
Child 5
Descendant of Root, Top-level consumer II, & Sub-consumer ii on consumer Branch B
Sibling of Child 4

Create your consumers

You have created a diagram mapping out your business structure. You have read the descriptions of the parts of the tree and understand how they interact. You are logged on to the Platform Management Console as a cluster administrator.

It is difficult to reorganize your tree structure once you have created it, so preparation and planning are key. You need to create a consumer for each label on your business structure that you created above. If you have created user accounts and know who the consumer administrator is for each top-level consumer, you can specify those user accounts during this procedure. If not, you can assign consumer administrators later.

1. Click Consumers > Consumers & Plans > Consumers.

   A list of existing top-level consumers displays. By default, you have ManagementServices, SampleApplications, and ClusterServices.
2. Create a consumer for your most important top-level business area.

   Use the plan you have already laid out and create a consumer with the name of that area. From the example above, the first consumer is "QA".
   a) Select Global Actions > Create a Consumer.
   b) Specify the name for your most important consumer.

      The most important top-level consumer in the plan above is “QA”.
   c) If you want, specify administrators and users, or assign them later.
   d) Specify an OS user account associated with this consumer.
   e) Leave the default resource groups selected.

      If you have already created more resource groups or replaced ComputeHosts with others, select as many as you would like available throughout this branch of the tree. If you do not select ManagementHosts and InternalResourceGroup resource group plus at least one other group, you have insufficient resources to run your applications on this entire branch.
   f) Leave the Reclaim behavior section blank.

      Reclaim behavior is an advanced feature and must be coordinated with the resource plan settings.
   g) For all consumers (except for those in the ManagementHosts and InternalResourceGroup resource groups), ensure the box Rebalance when time intervals change is checked within individual Consumer Properties dialog boxes.

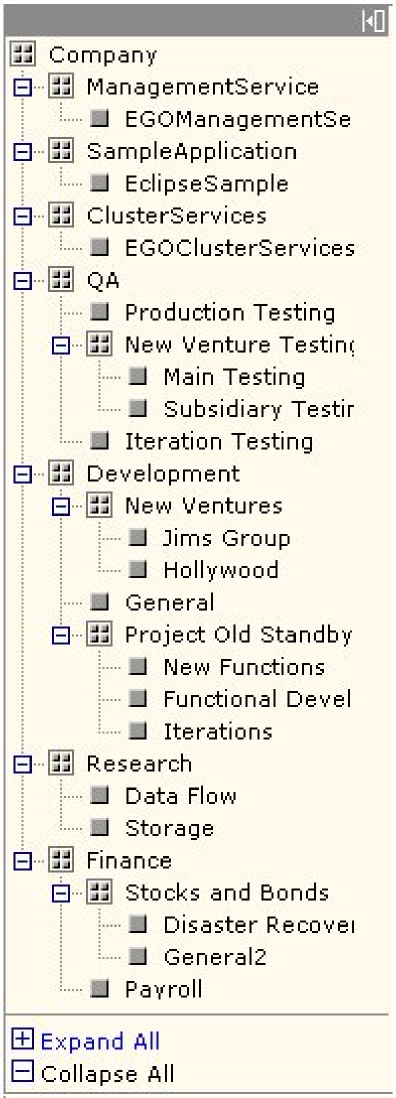
      This ensures that when your resource plan changes according to set time intervals, that originally configured share ratios, allocations, and lend/borrow policies are reapplied and enforced across all consumer branches in the consumer tree.
   h) Click Create.
3. In the order of the priorities you have already established, create the rest of your top-level consumers.

   For example, following the plan above, create Development, Research, and then Finance.
4. Once you have all your top-level consumers in the order that you want, in the Platform Management Console use the tree to navigate to your first top-level consumer (for example, “QA”).
5. Create all the sub-consumers for the most important top-level consumer (for example, “QA”) in the order that you have prioritized them.
6. Navigate to each sub-consumer and create required leaf consumers.
7. Repeat this process until each branch of the tree is complete and matches the plan you made.

The final consumer tree follows the same structure as the business structure.

Tip:

Do not name any two consumers with the same name. In the example below, the number “2” was added to the second consumer named “General” to distinguish them.



- Company
  - ManagementService
    - EGOManagementSe
  - SampleApplication
    - EclipseSample
  - ClusterServices
    - EGOClusterServices
  - QA
    - Production Testing
    - New Venture Testin
      - Main Testing
      - Subsidiary Testi
    - Iteration Testing
  - Development
    - New Ventures
      - Jims Group
      - Hollywood
    - General
    - Project Old Standby
      - New Functions
      - Functional Devel
      - Iterations
  - Research
    - Data Flow
    - Storage
  - Finance
    - Stocks and Bonds
      - Disaster Recove
      - General2
    - Payroll

How you are prepared

Now that you have a consumer tree that mirrors your business structure and is ordered according to the most important areas, you are prepared to go register applications to leaf consumers and customize your resource plan.

You are also ready to add user accounts and assign administrators to consumers.

Tutorial: Add a New Service to the Cluster

Goal

You plan, prepare the cluster for, and add four new services ensuring you allocate resources to run workload units.

Description

This scenario takes you through the steps of getting your cluster prepared for registering four new services, modifying your service profile, and registering your service. You have already created a basic tree that follows your business structure and you have either modified the default resource group ComputeHosts or you have decided to leave that resource group as-is (with all your compute hosts).

Suppose we have two different lines of business under a consumer branch called Research: Data Flow and Storage.

Data Flow

- Data Flow has one service: SuperMortgage
- Data Flow needs 30 CPUs

Storage

- Storage has three services: Fast, Slow, and SuperFast
- Fast needs 10 CPUs, Slow needs 50 CPUs, SuperFast needs 20 CPUs

At a glance

1. Gather the facts
2. Plan
3. Create consumer users
4. Create workload execution user accounts and ensure they exist on all hosts
5. Create the consumer structure
6. Change your resource plan
7. Register a new service
8. Run your service to see instances

Gather the facts

Before starting to register your services, organize the information you have. This is the most important part of the process—planning.

Pay special attention to the number of services, how many CPUs are required, and the operating system type of your machines.

Business process	Services	Host OS type	CPU slots required
Data Flow	SuperMortgage	Linux	30
Storage	Fast	Windows	10
	Slow	Linux	50
	SuperFast	Windows	20

Plan

Adding services requires creating consumers and changing the resource plan to ensure the new consumers have the necessary resources.

The checklists below provide guidance for the types of information that you need when actually creating a new service in the software.

How many services do you have?

- Number of consumers

  There must be one consumer per service.

  The consumer must be a leaf consumer—it cannot have any descendants.
- Free CPU slots on management hosts

  There must be one CPU slot free for each service on management hosts.

  - To identify free CPU slots in the cluster, use the resource plan in the Platform Management Console.
- Free CPU slots on compute hosts

  Are there enough CPU slots available on compute hosts to be allocated to the new services?

  The compute host resource group must have enough CPU slots available. Otherwise, you may need to add more hosts or shift resources around existing consumers.

  - To identify free CPU slots in the cluster, use the resource plan in the Platform Management Console.

Which user accounts are you going to use?

- Workload unit submission user accounts

  In order to submit workload units to the cluster, there must be user accounts defined in EGO that can connect and submit workload units. These accounts only exist in the EGO user database.
- OS user accounts

  Each consumer must be associated with an operating system user account (workload execution user account). A consumer's workload units run under this account.

  The user account must exist on all the hosts in the cluster, including management hosts. On Windows, it is recommended that these user accounts be domain accounts. Also, the password must be the same on all hosts.

Summarize your plan

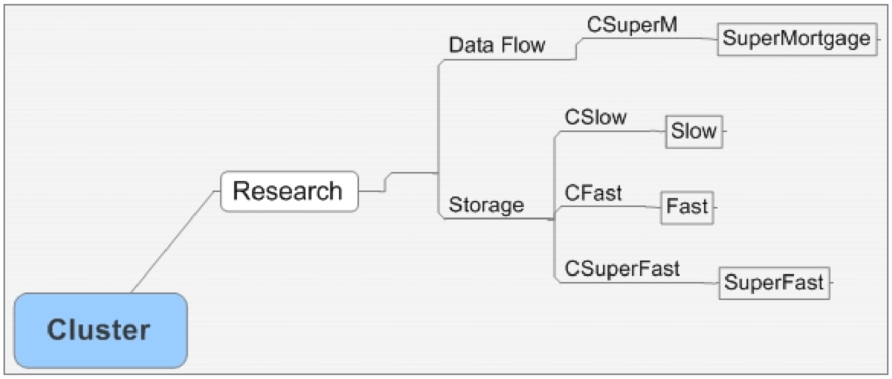
Have a summary of your plan so that you can refer to it when working and you can use it as a checklist.

Business process	Services	Leaf consumer name	Workload execution user account	CPU slots required on compute hosts to run workload units	CPU slots required on management hosts to run service
Data Flow	SuperMortgage	CSuperM	userm	30	1
Storage	Fast	CFast	userdf	10	1

Business process	Services	Leaf consumer name	Workload execution user account	CPU slots required on compute hosts to run workload units	CPU slots required on management hosts to run service
	Slow	CSlow	userds	50	1
	SuperFast	CSuperFast	userdsf	20	1

Draw your consumer structure

It helps to draw the consumer and service structure before you work in the software. Note that the leaf consumers are on the right, attached to a service.



Create consumer users

Through the Platform Management Console, create user accounts for your consumers that are able to submit workload units. You give these user accounts the role of consumer users when you create the consumer.

Create workload execution user accounts and ensure they exist on all hosts

When you create a consumer, you are required to specify an OS user account—this is the OS account under which workload units run. Ensure this account exists on all hosts in the cluster.

If you specify a Windows user account, and the user has not already been configured, you have to run `egosh ego execpasswd` to register the user account password with EGO before the execution user can run an activity without exiting.

Create the consumer structure

Create consumers and the consumer structure that you have planned out.

Note that the consumers that are to be attached to services must specify hosts from both a resource group that contains compute hosts (ComputeHosts by default) and the ManagementHosts resource group.

Use your plan summary to enter the required information for the consumer.

Create consumers

When creating a consumer, you need the following information:

- User account for consumer user
- User account for consumer administrator
- OS user account (and domain if Windows or a mixed cluster)
- Resource group name(s) for compute hosts

1. In the Platform Management Console, click Consumers > Consumers & Plans.

   A list of existing top-level consumers in the consumer tree displays.

   For example, CSuperM (the first leaf consumer) is under Data Flow according to the plan above. On the tree, click Data Flow. A message displays, indicating "'DataFlow' is a lowest level consumer. Currently, no applications are registered to it, so subordinate consumers can be created".
2. In the consumer tree, locate and click the consumer for which you would like to add a sub-consumer.
3. Select Global Actions > Create a Consumer.
   a) Specify the consumer name.

      For example, create a consumer with the name CSuperM.
   b) Review consumer administrators.

      A consumer administrator is pre-selected if this is a lower level consumer. Consumer administrators are only specified at the top-level consumer; they get inherited down through their branch.

      If you have not specified a consumer administrator for this branch, the default is the cluster administrator.
   c) Add a consumer user.

      Consumer users can monitor and control only the workload units they are assigned to.
   d) Specify an OS user account.

      The OS user account is the account under which all workload runs.

      If you specify a Windows user account, include the domain. For example `mydomain\userA`.
   e) Specify one or more resource groups this consumer should have access to.

      At minimum, select the ManagementHosts resource group to run the service. If you intend to run workload units for this consumer, select at least one other resource group for the service instance to run the activity. Only the resource groups specified by the parent consumer in the tree are available for selection. If you have not modified your resource groups, you can keep the default resource group selections.
   f) (Optional) Specify a reclaim grace period.

      If you intend this consumer to have a service that runs vital workload units that cannot be interrupted, specify the average length of a workload unit as the reclaim period. This only applies if you have set this consumer to borrow resources from other consumers (an advanced feature).
   g) For all consumers (except for those in the ManagementHosts and InternalResourceGroup resource groups), ensure the box Rebalance when time intervals change is checked within individual Consumer Properties dialog boxes.

This ensures that when your resource plan changes according to set time intervals, that originally configured share ratios, allocations, and lend/borrow policies are reapplied and enforced across all consumer branches in the consumer tree.

h) Click Create.

4. Continue creating the consumers you need for your services.

   For example, under Storage create three consumers: CSlow, CFast, and CSuperFast.

Now that you have your infrastructure in place for the services you plan to register, you must make sure you allocate enough resources to the new consumers.

Change your resource plan

Change your resource plan to allocate resources for the new consumers.

Refer to your plan summary that you already created.

Allocate resources for the consumer created for the service

You must be a cluster administrator or consumer administrator to perform this task.

When allocating resources, you need to ensure there are CPU slots are available from a resource group that is not ManagementHosts.

1. In the Platform Management Console, click Consumers > Consumers & Plans.
2. In the consumer tree, click the cluster name. (Resource plans are set at the cluster level.)

   A list of consumers that make up the cluster displays.
3. Click Resource Plan.
4. From the resource group menu, select ComputeHosts (or equivalent).

   If you have already modified your default ComputeHosts resource group or have created new resource groups, select the resource group that has the compute hosts you want.
5. Click Expand All at the bottom of the page to expand the resource group tree.

   Only the consumers that have the ComputeHost (or equivalent) resource group selected in their properties display. If the consumer you want does not display, then it does not have this resource group selected in its properties.
6. Specify the number of CPU slots required to perform computations on compute hosts for your consumer in Owned Slots.

   For example, the plan says that the SuperMortgage service needs 30 CPUs. In the resource plan for the consumer CSuperM (the leaf consumer that will have SuperMortgage registered to it), add **30** to Owned Slots.

   If your balance shows a deficit of CPUs, this means you allocated more CPUs to a leaf consumer than were available through its parent. Adjust the ownership throughout the entire branch until the balance is 0.

   If you need more CPUs than are ever available, add hosts to your cluster or adjust your resource groups.
7. From the resource group menu, select ManagementHosts.
8. For your consumer, specify 1 owned CPU slot from the ManagementHosts group.

   The 1 owned CPU slot that you specify is used to run and schedule workload.

Since it is essential that this CPU slot be available for the service when it is required, it is essential to own the 1 CPU slot. Do not enable lending.

9. When you are satisfied with your resource allocation and your balance is zero or positive, click Apply to make it active or Export to save it for later use.

Register a new service

You must be a cluster administrator to perform this task.Registered services may need to be enabled afterwards if the service is set up to be manually started.

1. From the Cluster page, click Monitor Services.

   The Services page displays.
2. In your consumer tree, navigate to the location where you want to register your service (for example, SuperMortgage).

   You can only register a service to a leaf location. If your consumer has any sub-consumers, then it is not a leaf.
3. From Global Actions, select Register a new service.

   The Service Profile dialog displays showing the minimum required and preferred attributes for a new service, along with preconfigured default values.
4. Click Table Preferences at the top of the dialog box, and check all boxes to add the corresponding information columns to the table.
5. Modify the service profile with service-specific details.
   a) In the sc:ServiceDefinition section, click in the ServiceName value field and type new name for the service (for example, **SuperMortgage**).

      The name can only contain up to 40 characters, and can contain letters, numbers, and underscores ("_") . The first character of the name cannot be a number.
   b) Insert the parameter for ResourceGroupName and then specify the resource group from which compute hosts are to be selected for the consumer, following these steps:
      1. In the sc:sc:AllocationSpecification > sc:ResourceSpecification section, click Actions.
      2. From the drop-down list, select Insert "ego:ResourceGroupName".
      3. In the ResourceGroupName row, type a resource group name in the value field.

         If you have not created new resource groups, specify the default resource group ComputeHosts. Note that the resource group name must exactly match the compute host resource group specified in the consumer properties.
   c) In the sc:ActivityDescription > ego:ActivitySpecification section, select Insert "ego:ExecutionUser" from the drop-down list, and then specify the OS execution user account that runs the service (for example, **egoadmin**).
6. Click Register.

Your service is registered. You may see the state change between DEFINED, INIT, ALLOCATING, and STARTED.

If your state remains DEFINED, you need to start your service.

Run your service to see instances

If you want to run your service from a machine that is not part of the cluster, you need to configure the machine to connect to the cluster.

1. Run your service.
   a) From the Cluster page, click Monitor Services.

      The Services page displays.
   b) In your consumer tree, navigate to the location where the service you want to start is registered or see a list of all registered services by clicking the cluster name.
   c) For the service you want to start, select Actions > Start.
   d) Click OK to confirm that you want to start that service.
2. Use the Platform Management Console to view service instances.
   a) From the Services page, locate your service.
   b) Click the name of your service, and then click Service Instances.

      A list of service instances display.
   c) For details about any service instances, click the service instance Sequence number.

      Service instance properties display.
3. Repeat the process for all your new services.

Scenario: Creating a resource plan that responds dynamically to consumer needs

Goal

In order to break down self-contained resource group "silos", and instead move to a more distributed sharing model, you need to create an effective resource plan that enables lending and borrowing. Following the steps below, you create a hybrid resource plan that promotes and protects the business objectives of various consumers yet still allows resources to be shared between them over your cluster.

Description

You have created useful resource groups, set up a consumer tree that mirrors your business structure, and added various business services to your cluster. You now want to create resource plans for the consumers in your consumer tree. If you are a cluster administrator, you can define a distinct resource plan for a time window of significance for each consumer.

Specifically, you want to create resource plans that ensure the following:

- Your top-priority consumer (QA) receives the greatest number of resources, has first claim on any unowned resources, and is first in line to borrow available resources if demand is high.
- Development, Research, and Finance consumers follow QA in priority sequentially, have appropriately adjusted ownership allocations, and have lending and borrowing policies that compliment each other's business objectives.
- Research receives extra protection against failures that might occur during a critical back-up window between 2:00 AM and 3:00 AM each day.
- Finance moves up in priority for the last 3 days of each month during a time of high activity.

You therefore want a resource plan that allows for sharing between these consumers, but that guarantees certain consumers have a higher priority than others in accessing these resources during known peak periods of activity.

At a glance

1. Why share?
2. Before you create a resource plan
3. Create a resource plan with different time intervals
4. Create and import a distinct resource plan for use at the end of each month

Why share?

Novice EGO administrators may implement a "silo" model for initial testing, creating resource groups that remain quite distinct and self-contained, and configuring resource plans to exclude the ability to borrow and lend resources between consumers. By maintaining such a limited model, administrators lose the advantages of EGO's flexibility to dynamically respond to consumer and client needs and to distribute resources effectively across a cluster. To break down the silo and move to a more distributed sharing model, you must enable lending and borrowing. This can be done by altering the resource plan in a variety of ways, and ranking consumers.

Before creating a resource plan, you must first understand its components and capabilities.

Before you create a resource plan

1. Go to your ManagementHosts resource group and ensure there is at least 1 owned slot allocated to each registered leaf consumer.

   Unless you have at least 6 or 7 owned slots allocated to the ManagementHost resource group, EGO services does not run (there are 6 out-of-box management services installed, each requiring its own slot; if derbydb service was enabled during installation, then there are 7 installed services). If you have other non-EGO services registered, you may require additional allocated slots.
2. For all consumers (except for those in the ManagementHosts and InternalResourceGroup resource groups), ensure the box Rebalance when time intervals change is checked within individual Consumer Properties dialog boxes.

   This ensures that when your resource plan changes according to set time intervals, that originally configured share ratios, allocations, and lend/borrow policies are reapplied and enforced across all consumer branches in the consumer tree.
3. Ensure that the resource plans you create apply to consumers that are outside of the ManagementHosts and InternalResourceGroup resource groups (be sure to change resource groups in the Console before you start working on a resource plan).

   To do this, navigate to Consumers > Consumers & Plans > Resource Plan, and then click Resource Groups: ComputeHosts (or equivalent) from the drop-down list.

   This is important so that you do not interrupt services inadvertently when invoking different resource plans with different time intervals. The services within the ManagementHosts and InternalResourceGroup resource groups should run without interference regardless of which policy is applied during a certain time of day.
4. (Optional). You may wish to work through other related scenarios. For the sake of continuity, some examples are reused in this document.

Create a resource plan with different time intervals

Creating a resource plan can be a complex process. This process has been broken up into 7 major steps that take place in sequence, all within the Resource Plan page of the Platform Management Console.

A simple default resource plan is already in place to adapt for your own purposes. Any changes you make and apply to the resource plan in the Platform Management Console are implemented immediately. Do not apply the changes if you do not wish the changes to be immediate. Export the plan instead, and work on the copy until you are ready to import the updates and apply the changes.

1. Create a time interval
2. Set owned slots for leaf consumers
3. Set lend and borrow policies
4. Specify share ratios for resource distribution between sibling leaf consumers
5. Set consumer rank to establish lending priority between sibling leaf consumers
6. Create a different resource plan for another time interval
7. Apply or export resource plan

Create a time interval

1. Navigate to Consumers > Consumers & Plans > Resource Plan.

   If you have never updated or imported a resource plan, your resource plan shows default settings.
2. If you have not already done so, select the resource group for which you are creating a resource plan (for example, ComputeHosts).

   You do not want to change the resource plan for ManagementHosts or InternalResourceGroup (the resource groups where EGO services reside). If you are working from a default plan, select Resource Groups: ComputeHosts (or an equivalent resource group you may already have created) from the drop-down list at the top of the resource plan.
3. Select Time Intervals and Settings > Insert a Time Interval.
4. Select the start time 02:00 and the end time 03:00. Click Insert.

   A confirmation dialog opens.
5. Click OK to continue.

   The plan now contains the following time intervals: 00:00-02:00, 02:00-03:00, and 03:00-24:00.

You have created the time window during which the consumer named Research performs critical back-ups each day. Separate resource plans need to be created for each of the three time intervals. These resource plans automatically get applied during these times.

Continue creating your resource plan by assigning slots to consumers.

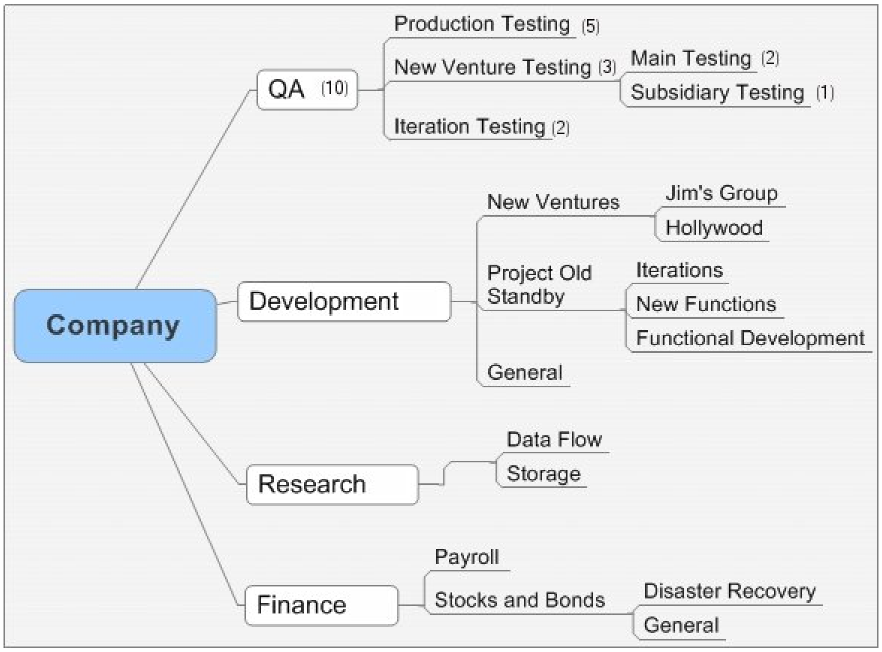
Set owned slots for leaf consumers

1. Click Expand All to expand the entire consumer tree.

   You only see those consumer branches that specify as part of their consumer properties the resource group you are currently creating a resource plan for (in this example, ComputeHosts).
2. Select Time Intervals and Settings > Show Advanced Settings to display advanced settings and policy configuration options for the resource group.
3. For the first time interval only (00:00-02:00), set the owned slots for each leaf consumer in the QA consumer branch.

   Be sure to allocate all available resources. When all resources are allocated, the Balance at the bottom of the Ownership section is 0.

   For example, if you have 10 owned QA slots to distribute, you may want to distribute 5 of them to the “Production Testing” consumer, 3 to the “New Venture Testing” consumer, and 2 to the “Iteration Testing” consumer (see the graphic below). If one of your consumers has sub-consumers, ensure that you distribute owned resources to them as well. For example, if the “New Venture Testing” consumer has 2 sub-consumers, you might assign 2 of the 3 available slots to the “Main Testing” sub-consumer, and 1 slot to the “Subsidiary Testing” sub-consumer.



Note:

Keep in mind that the resource plan is hierarchical in nature; you cannot allocate more resources to sub-consumers than the consumer above it owns. Be sure to distribute all available resources as far down a consumer branch as possible to its leaf consumers.

Continue creating your resource plan by configuring consumer lending and borrowing policies.

Set lend and borrow policies

Lending allows a consumer’s unused slots to be used by other consumers. Borrowing lets a consumer use unowned or lent slots when they are available.

1. For the first time interval only (00:00-02:00), select the options to Lend and Borrow for each leaf consumer in the QA branch that has something registered to it.

   In our example, the leaf consumers (consumers with no descendents) in the QA branch include the following:

   - Production Testing
   - Iteration Testing
   - Main Testing
   - Subsidiary Testing
2. Specify the maximum number of slots you would like each consumer in the QA branch to lend.

   The lend limit number is a maximum, absolute number of slots to be lent out during periods of non-use. The remainder are never lent, even if they are not being used. If you leave the field blank, all slots can be lent out.

   For example, beside the Lend field for the leaf consumer Production Testing, click Details. From the Lend Details dialog box, enter the number **5** in the Cumulative lend limit field (or leave the field blank) to indicate that this consumer is willing to lend out all 5 owned slots during periods of non-use. If you choose to make only 4 slots available for lending instead of all 5, this consumer always retains 1 resource that can never be borrowed.

In our example, leave all lend limit fields blank for all the leaf consumers in the QA branch. This promotes a more distributed sharing model for your cluster.

3. Specify particular consumers to lend to.
    a) For each leaf consumer in the QA branch that has lending enabled, click Details beside the consumer’s Lend | Limit box.

       For example, click Details for the consumer Production Testing.

       The Lend Details dialog box opens for this consumer.
    b) Check the boxes beside those consumers Production Testing lends to. In our example, choose only consumers within the QA branch of the consumer tree.

       Tip: It is useful to indicate which consumers to lend resources to if you want to ensure that unused resources are only lent to consumers within a specific consumer branch.
    c) Click Apply, and then Close.
    d) Repeat these steps for the remaining leaf consumers in the QA branch.
4. Specify the maximum number of slots a consumer can borrow from lending leaf consumers in the QA branch.

   This number is the maximum, absolute number of slots that can ever be borrowed by a consumer. If you leave the field blank, borrowing is limited only by the amount of resources available in the cluster. For our example, do not specify limits—leave the fields blank.

   For example, beside the Borrow check box for the leaf consumer Production Testing, leave the Limit field blank. This allows the Production Testing consumer (within the highest priority QA consumer branch) to borrow as many resources as needed to meet its resource demands.
5. Specify particular consumers that you want each consumer to borrow from, along with the order in which they borrow from those consumers.
    a) For each leaf consumer in the QA branch that has borrowing enabled, click Details beside the consumer’s Borrow | Limit box.

       For example, click Details for the consumer Production Testing.

       The Borrow Details: Production Testing dialog box opens.
    b) Check the boxes beside those consumers Production Testing can borrow from. In our example, choose all consumers that have lending enabled within the consumer tree (even those from other consumer branches).

       Tip: It is useful to indicate which consumers to borrow resources from if you want to ensure that unused resources are only borrowed from consumers within a specific consumer branch.
    c) Use the Preference Order column to specify the order in which you want to borrow from specific consumers.

       The number **1** indicates that this consumer is your first choice to borrow from. Do not repeat any numbers.

       In our example, indicate a higher borrowing order for consumers in other consumer branches (outside of the QA consumer branch). This helps to ensure that your highest priority consumer (Production Testing) within your highest priority business process (QA) can borrow from as many consumers as possible during periods of peak activity.

Set your borrowing order in this way:

- QA
  - Production Testing (not available for ordering)
  - New Venture Testing
    - Main Testing: Preference Order **14**
    - Subsidiary Testing: Preference Order **13**
  - Iteration Testing: **12**
- Development
  - New Ventures
    - Hollywood: Preference Order **11**
    - Jims Group: Preference Order **10**
  - General: Preference Order **9**
  - Project Old Standby
    - New Functions: Preference Order **8**
    - Functional Dev: Preference Order **7**
    - Iterations: Preference Order **6**
- Research
  - Data Flow: Preference Order **5**
  - Storage: Preference Order **4**
- Finance
  - Stocks and Bonds
    - Disaster Recovery: Preference Order **3**
    - General: Preference Order **2**
  - Payroll: Preference Order **1**

d) Click Apply, and then Close.

e) Run `egosh ego restart` to have the changes take effect.

f) Repeat these steps for the remaining leaf consumers in the QA branch.

Continue creating your resource plan by setting share ratios for consumers on the same branch of the consumer tree.

Specify share ratios for resource distribution between sibling leaf consumers

The share ratio determines the minimum number of resources to distribute to a leaf consumer in cases where there is competition between siblings.

In our example, a set share ratio applies between the leaf consumers Main Testing and Subsidiary Testing (because they are sibling leaf consumers with the same parent on the QA branch), but does not apply between Main Testing and Iteration Testing (because these are leaf consumers on different levels of the QA branch and not siblings).

1. Check the Share Ratio box for the leaf consumers Main Testing and Subsidiary Testing in the QA branch, and enter the share ratio **1** for each of them.

   This ensures that both consumers share available surplus resources equally.
2. Check the Share Ratio box for the consumers Production Testing, New Venture Testing, and Iteration Testing, and then enter the share ratio of **2**:**1**:**1** respectively.

This ensures that the higher priority consumer Production Testing receives twice as many unused resources as either of the leaf consumers New Venture Testing and Iteration Testing from the same branch. New Venture Testing shares resources equally with Iteration Testing.

Furthermore, New Venture Testing passes on the shared resources it receives directly to its leaf consumers.

3. Indicate a sharing limit for leaf consumers.

   By default, a leaf consumer who is first in line to borrow (either because it is the first one to express a demand, or because of its higher consumer rank), can borrow all the resources available for sharing if it has enough demand. If you want to limit the amount of resources a consumer receives through borrowing, you can indicate a limit in the Sharing Limit field. The borrowing limit is expressed as a numeric quantity.

   In our example, do not specify a borrowing limit for leaf consumers in the QA branch. Because they are high priority consumers, you want to ensure that their demands for resources are fully satisfied in order to fulfill business objectives.

   Note: If the consumer owns resources in addition to borrowing resources, the maximum distribution to the consumer is the borrowing limit plus the ownership.
4. Repeat these steps for the other consumer branches, establishing an equal share ratio between sibling leaf consumers, and setting sharing limits for lower priority consumers.

Continue creating your resource plan by assigning consumer rank.

Set consumer rank to establish lending priority between sibling leaf consumers

If you have critical workload units to run, you can ensure resources are available by setting a high priority to a consumer.

1. Under Ownership, rank consumers within the QA branch.

   Specify any positive whole number in the Consumer Rank field, where 1 is the highest priority. Priority settings are relative to one another within the resource group. If you leave the priority blank, that consumer has no priority over any other consumer (it does not form part of any consumer ordering/sequencing).

   Note: Setting a high priority to all consumers effectively nullifies any advantage. Only set consumer rank high for selected consumers.

   In our example, give Production Testing a high ranking value, such as 50. Rank other leaf consumers in the QA branch high as well, although with a lower value than you gave to Production Testing.
2. Do this for other consumer branches, assigning decreasing consumer rank values as you move down the consumer tree.

   For low priority consumer branches, such as Finance, you can choose to leave the Consumer Rank field blank.

Note:

Plan the ranking of your consumers knowing that resources are reclaimed based on their rank; those leaf consumers with a lower consumer rank are reclaimed before consumers with a higher rank.

Continue creating your resource plan by configuring another resource plan for the next time interval. If you wish to export the plan at this time and continue creating it later, skip to the topic Apply or export resource plan.

Create a different resource plan for another time interval

At this point you have created a complete resource plan for the time interval 00:00-02:00. Your top-priority consumer branch (QA) and it's sub-consumers are set to receive the greatest number of available resources, have first claim on any unowned resources, and are first in line to borrow available resources if demand is high.

Between 02:00-03:00, a traditionally lower-priority consumer (Research) requires immediate access to more resources in case it experiences host failure during this critical time window. A new resource plan is therefore required during this time interval that favors the Research branch in terms of resource allocation and borrowing policies.

1. When setting owned slots for leaf consumers in the Research consumer branch, ensure all resources are allocated to leaf consumers.
2. When setting lend and borrow policies for the Research consumer branch, do the following:
    a) For the Research consumer branch, enable lending.
    b) Specify that leaf consumers in the Research branch only lend to each other, not to other consumers from other branches.
    c) Enable borrowing with no limits.
    d) For consumers from other branches, specify the leaf consumers from the Research branch as approved borrowers.
    e) For consumers from other branches that are specifying the order in which they want to borrow from other consumers, give the leaf consumers from the Research branch a high numerical value (to indicate they are not a number 1 choice to borrow from).
    f) Run `egosh ego restart` to have the changes take effect.
3. When setting the share ratio for the Research branch, specify an equal share ratio to all leaf consumers (enter the value **1** in all fields).
4. When setting the consumer rank for each consumer, assign lower numerical values to consumers from the Research consumer branch.

    The lower the number, the higher the rank; therefore, a “1” indicates the highest ranking.

You have now created two resource plans: The first plan configures the QA consumer branch to be a top-priority consumer who qualifies to receive the greatest number of resources, has first claim on any unowned resources, and is first in line to borrow available resources if demand is high. The second plan ensures that the Research consumer branch receives extra protection against failures during a critical back-up window each day.

You must create a third resource plan for the final time interval, 03:00-24:00. Mirror the same plan that was created for the original time windows 00:00-02:00.

You must also repeat these steps for any other resource groups you have created. (Do not alter the plan for the ManagementHosts resource group.) To switch between resource groups, use the drop down list above the plan. Make sure you apply changes before switching to a different resource group.

Apply or export resource plan

Any changes you make and apply to the resource plan in the Platform Management Console are implemented immediately.

1. Click Apply to save and make the current settings active.

   If you do not want to make the current changes active, export the resource plan instead.
2. If you do not want to immediately apply the resource plan you have created, you can export it for later use.
   a) From Consumers > Consumers & Plans > Resource Plan, go to the bottom of the page and click Export.
   b) Save your resource plan locally.

      You may want to give the XML file a unique name including date or plan specifics so that you can import the plan of your choice quickly and easily at a later time.

Note:

Do not change your tree structure while the plan is exported or you will not be able to import the plan again.

Once you apply the resource plan, it becomes active. You can export this plan and import it again as needed if you want it to apply to specific days of the week only. You can create as many resource plans as you want and import them as needed for a quick change in your distribution.

Create and import a distinct resource plan for use at the end of each month

Create a resource plan that favors the consumers in the Finance branch of your consumer tree, and ranks them with a high consumer priority. Ensure that the critical backup time window from 02:00-03:00 is preserved to protect the consumers in the Research branch.

The Platform Management Console stores only your active resource plan. You can, however, import and apply a previously created resource plan (saved in XML format).

For the last three days of each month, the consumers in your Finance branch move up in priority during a time of high activity for them. You need to import and apply and different resource plan for this time.

1. Ensure that a resource plan exists (and was exported) that favors the consumers in the Finance branch of your consumer tree, and ranks them with a high consumer priority.
2. Ensure that the resource plan you import has an identical tree structure to the active resource plan.

   It is ok if the time intervals are different.
3. Click on your cluster name in your tree.
4. Click Consumers > Consumers & Plans > Resource Plan.

   If you do not see Resource Plan, make sure you are at the top of your tree.
5. At the bottom of the page, click Import.

   The Import a resource plan (XML) dialog displays.
6. Browse for the location of the XML resource plan you wish to import.
7. Click Import.

   Importing a resource plan makes it the active plan.

Scenario: Creating a resource plan that responds dynamically to consumer needs

You have imported a resource plan and made it the active plan. All distribution of resources take effect immediately.

Scenario: Changing resource distribution models

Goal

You change your current resource distribution model (from either full ownership with lending and borrowing or full share with no ownership) to a hybrid model that supports both resource ownership and sharing.

Scenario A: From Ownership to Hybrid model

As a cluster administrator, you have originally set up a resource distribution model that supports full resource ownership. You may have enabled lending and borrowing of resources within the cluster, or you may have set up a “silo” model where there is no borrowing or lending of resources between consumers.

You now wish to change models so that consumers can have access to both owned resources (either their own or those allocated to other consumers) and shared cluster resources (unowned resources that make up the cluster’s share pool). You want a more flexible and less prescribed model of resource distribution that responds better to fluctuating workload conditions and number of cluster resources.

Scenario B: From Share to Hybrid model

As a cluster administrator, you have set up a distributed resource sharing model where there are no owned slots, but where each consumer has an assigned share ratio of cluster resources (“share pool”).

Your company has a new line of business that requires specific resources (for example, from a certain resource group). The previous full share model, where resources were flexibly allocated depending on workload conditions, consumer priority, and resource availability, cannot guarantee the allocation of the hosts required to satisfy the needs of the new line of business. You want to implement a hybrid model where some resources remain in the share pool, while others are allocated specifically to the new line of business in order to guarantee a minimum level of resource allocation.

At a glance

1. Gather the facts
2. Plan
3. Change the resource plan to support a Hybrid model
4. Add a new consumer to a Hybrid model resource plan
5. View the number of owned and shared slots

Gather the facts

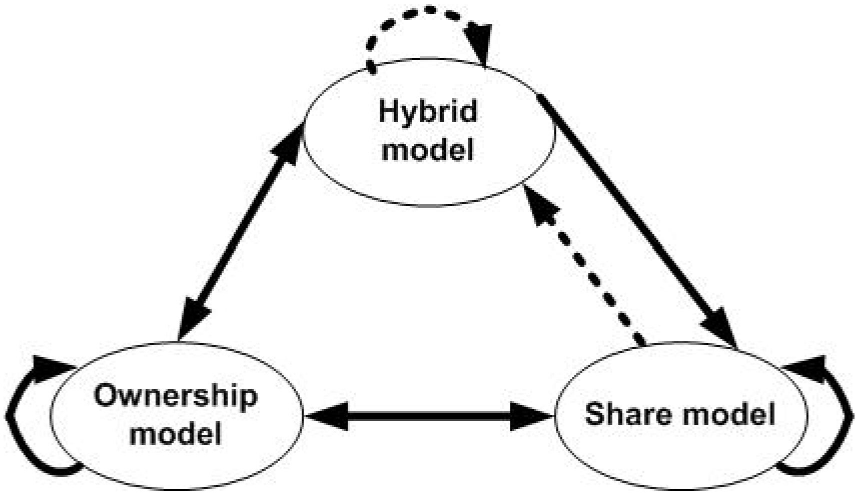
EGO supports three resource distribution models in the resource plan.

Ownership model (“silo”, or with lending/borrowing)	Share model	Hybrid model
This model can support resource distribution ranging from “silo” (where each consumer has a fixed ownership number and there is no lending and borrowing between them), to limited lending/borrowing among consumers, to unlimited lending and borrowing of resources between consumers. **Note:** Share limits are set to zero for all consumers in this model.	In this model, each consumer has a portion (assigned share ratio) of the clusters resources (“share pool”). Each consumer has the potential to use resources from the share pool if other consumers have no competing demand for resources. If there is competition between consumers for resources from the share pool, note the following: **1.** A consumer’s rank is enforced. A consumer with a higher rank than its sibling receives its ratio of share pool resources first. **2.** A consumer’s assigned share ratio is enforced. EGO automatically reclaims the resources from the over-allocated consumer. **Note:** Consumers do not own any slots in this model. Set ownership to 0, as lending and borrowing configurations are of no effect. Ensure share ratios are suitably configured across all consumer branches and for each consumer.	In this model, you can create a resource plan that combines both models (Hybrid model). In this model, EGO maintains a share pool and also supports configured resource ownership by leaf consumers.

The following diagram shows all resource distribution models and potential model migrations. Each arrow represents either a complete migration from one model to another (there are six migration possibilities) or a change within a current resource distribution model.

Note:

The dotted arrows indicate those resource distribution migrations/changes with notable restrictions. Resource distribution model migrations/changes indicated by solid arrows have no restrictions or associated issues.



The are restrictions for those resource distribution model migrations/changes indicated by dotted arrows.

Restrictions when migrating from a Share to a Hybrid model

When you migrate from a Share model to a Hybrid model, resources that were previously in the share pool are allocated as owned resources to consumers in the resource plan; the number of share pool resources therefore decreases, which may potentially cause difficulties.

In some cases, you may not be able to immediately satisfy the ownership requirements set out in your resource plan when migrating from a Share to a Hybrid model. When the new resource plan comes into effect, and the unallocated share pool resources (in the original Share model) are fewer in number than the configured ownership number in the plan (in the new Hybrid model), then the resource distribution requirements of the plan cannot be fulfilled.

Example: In the Share model, the share pool has 100 slots: 60 slots (resources) are already allocated to consumers running workload units, and 40 slots are available for allocation. Upon implementing a Hybrid model, you intend to reduce the share pool to 50 slots and distribute the remaining 50 slots to three consumers (consumers A, B, and C) with an ownership allocation of 20, 20, and 10 respectively. But after reconfiguring the resource plan, consumer C cannot get its 10-slot allocation. This occurs because there are resources in the share pool that are already allocated. During a resource plan migration from a Share to a Hybrid model, the normal logic for reclaiming share pool resources is not enforced.

Summary: If there are not enough unallocated resources in the share pool at the moment the new Hybrid plan takes effect (during runtime), then consumers are not allocated the planned number of owned resources. Allocated resources from the share pool must first be released back to the share pool by the clustered application managers that are using them (for example, Platform Symphony or LSF) before they can be reallocated as owned resources to a consumer.

Restrictions when making resource plan changes within a Hybrid model

When modifying a resource plan for an existing Hybrid model (where there are both owned resources and unowned share pool resources), there are some issues that may arise.

- If you add a new leaf consumer and allocate resources to it by taking them from the share pool, the consumer’s guaranteed ownership allocation may not be satisfied if the share pool resources are in use by other consumers. If there are not enough unallocated share pool resources to fulfill the ownership requirements in the new resource plan, then the new consumer does not receive its planned resources until sufficient resources are released back into the share pool.

- If you allocate share pool resources as owned resources to a consumer at any level in the consumer tree (top-level, parent, or child), and there are insufficient unallocated share pool resources to fulfill the ownership requirements of the new resource plan, consumers throughout the branch remain unsatisfied. (Top-level consumers are satisfied first with available resources, according to the plan, followed by parents and their children.)

There are no issues if resources taken from the share pool are available (if they are not currently allocated to other consumers).

These use-cases do not have any restrictions or associated issues:

- Adding a new consumer with ownership without changing the number of resources owned by the parent
- Reducing the number of resources owned by a consumer and increasing the share pool or changing the share ratio
- Removing some consumers

Plan

Before beginning, learn how to work with and create a resource plan. For more information, you may want to work through a related scenario Creating a Resource Plan that Responds Dynamically to Consumer Needs, as it contains detailed information and procedures on setting up a hybrid resource plan.

Change the resource plan to support a Hybrid model

You already have a working resource plan that supports either a Share or Ownership model. You may wish to export the existing resource plan and save it for later re-use before beginning your modifications.

When implementing a Hybrid resource plan, you must be sure there are enough unallocated resources in the share pool at the time that the new resource plan becomes effective. Unless specified, each step in the following task relates to migrations from either Share or Ownership models.

1. (Original model type: Share) To reclaim allocated share pool resources, and thereby increase the number of resources available for ownership in the plan, set the Limit in the Model Type: Share section of the Console to **0** for all consumers (Consumers > Consumers & Plans > Resource Plan).

   This forces the allocated resources to immediately return to the share pool. You can then allocate reclaimed resources from share pool to a consumer.

   Note: If you take this approach, a clustered application manager running on EGO (for example, Platform Symphony) may have its workload units disrupted if it is using the share pool resources that get reclaimed. Create a new time interval to avoid this issue:

   a) In the Resource Plan, create at least one new time interval. You want a time interval for the existing Share model resource plan, an interval for the new Hybrid model resource plan, and a short interval (for example, 1 hour) to fall between the Share and Hybrid model plans.

      The short time interval provides a buffer between the two resource plans, and allows currently allocated share pool resources that are running workload units to be released

    back to the share pool before the Hybrid plan becomes effective. The short time interval must be longer than the grace period for resource reclaim.
    b) In the short time interval that falls between the Share and Hybrid plans, set the Limit in the Model Type: Share section to **0** for all consumers.
    c) In the time interval that follows, set the desired resource policies to support a Hybrid model of distribution.
2. Set or adjust owned slots for each consumer to guarantee a minimum allocation of resources.

   When all available resources are allocated, the cluster balance at the bottom of the Owned Slots section changes to 0.

   Important:

   If you are allocating resources from the share pool to a consumer, ensure there are enough unallocated resources in the share pool at the time that the new resource plan becomes effective to satisfy the consumer's configured ownership.
3. Set or adjust a lend policy for each consumer; enable lending for leaf consumers who own resources.
4. Set or adjust a borrowing policy for each consumer; enable borrowing for leaf consumers who own resources.
5. Set or adjust share ratios for each consumer.
6. Set or adjust the consumer rank for each consumer.
7. Click Apply when all changes the resource plan are complete and you are ready to immediately implement the new plan.

Add a new consumer to a Hybrid model resource plan

You already have a working resource plan that supports a Hybrid model. You may wish to export the existing resource plan and save it for later re-use before beginning your modifications.

1. Create a new consumer.
    a) From Consumers > Consumers & Plans > Consumers, navigate to the location in your consumer tree where you wish to add the new consumer, and then select Create a Consumer from the Global Actions list.
    a) Fill in the consumer properties.
2. Update the resource plan.
    a) Set or adjust owned slots for each consumer to guarantee a minimum allocation of resources.

       When all available resources are allocated, the cluster balance at the bottom of the Owned Slots section changes to 0.

       Important:

       If you are allocating resources from the share pool to the new consumer, ensure there are enough unallocated resources from the share pool at the time that the new resource plan becomes effective. Otherwise, the new consumer does not receive its configured ownership until sufficient resources are released back into the share pool.

   b) Set or adjust a lend policy for each consumer; enable lending for leaf consumers that own resources.
   c) Set or adjust a borrowing policy for each consumer; enable borrowing for leaf consumers that own resources.
   d) Set or adjust share ratios for each consumer.
   e) Set or adjust the consumer rank for each consumer.
3. Click Apply when all changes the resource plan are complete and you are ready to immediately implement the new plan.

View the number of owned and shared slots

For comparison purposes, you can view the number shared slots that a consumer gets from the share pool along with the number of owned slots. Do this through the Console or using the command line.

1. From the Console, click Resources > Monitor Resource Allocation.
2. From the command line, run
   ```
   egosh consumer view consumer_name
   ```
   .

Index