ORACLE

GRID INFRASTRUCTURE

An Oracle Reference Guide
October 2019

Oracle Grid Infrastructure Agents

Version 10

11.2.0.4 and later
12.1.0.1 and later
12.2.0.1 and later
18 and later
19 and later
20 and later

Table of Contents

High Availability Components for Applications

Oracle Grid Infrastructure provides the necessary components to manage high availability (HA) for any business critical application. Oracle Grid Infrastructure Agents (XAG) are Oracle Grid Infrastructure components that provide the HA framework to application resources and resource types managed through the agent management interface, AGCTL. This framework provides a complete, ready to use application HA solution that contains pre-defined Oracle Grid Infrastructure resource configurations and agents to integrate applications for complete application HA.

Oracle Clusterware[1], a component of the Oracle Grid Infrastructure, provides a pre-configured public core network resource, ora.net1.network, to which applications can bind application Virtual IPs (APPVIPs) for network connectivity HA. Administrators can easily create additional network resources on distinct network interfaces, as required, for consolidation, network isolation and scalability. Oracle Grid Infrastructure will manage the network stack availability as well as all Oracle Grid Infrastructure resources with defined dependencies on the root network resource.

Applications require shared storage for binary, trace, metadata and state files. Shared storage enables the application to run on any node in the cluster by maintaining a single copy of binaries & configuration metadata and provides a persistent state tracking and restoration mechanism in the event of failover. Any Oracle certified shared or clustered file system can be used to satisfy the application shared storage requirements. This includes clustered file systems and failover file systems. To manage failover of OS native file systems, administrators will have to build the necessary script agent using the Oracle Clusterware script agent API. The Oracle Grid Infrastructure ships with the Automatic Storage Management Clustered File System (ACFS) which administrators can easily configure and present to the cluster for these purposes. An ACFS file system will be mounted on all user defined nodes and file system availability is managed entirely by the Oracle Grid Infrastructure. To enhance and automate application availability, applications can define start/stop dependencies on the ACFS configured file system resource, 'ora.<diskgroup>.<volume>.acfs'[2], so that the application is started, stopped or relocated as the ACFS resource comes online or goes offline. ACFS is the recommended HA file system for applications configured for Oracle Grid Infrastructure HA.

The Oracle Grid Infrastructure Agents (bundled or standalone) leverage these infrastructure components and allow for a simple integration which eliminates the need for supplementary infrastructure agents for network, IP and file system HA. In addition, the Oracle Grid Infrastructure provides rich resource modeling which applications simply plug into. Each application entity is represented and managed as a resource within the Oracle Grid Infrastructure. This resource modeling provides fine-grained resource state reporting, inherent placement policies, and pre-configured resource dependencies for the bundled agents.

These agents will also leverage existing databases and database services for a tighter dependency between the application and the database providing an even greater end-to-end availability model. Applications that connect to an Oracle Database and run in the same cluster can define dependencies on the database. If you configure a database dependency, the database will start up before the application so that it is open and available to the application when needed. For applications that connect to the database using the database SID, these applications

[1] Please see http://oracle.com/goto/clusterware for more details and links to specific Oracle Clusterware documentation

[2] Please see http://docs.oracle.com/cd/E11882_01/server.112/e10500/asmfilesystem.htm

should configure the '*--databases'* dependency. This will ensure that the application will run on the same node as the database. For applications that use TNS services to connect to the database, and thus can run on any node in the cluster and do not have the co-location requirement with the database or database instance, these applications should define their dependency on a *'--db_ services'*. This will allow, if so configured, the application and database to run on separate nodes in the cluster. This deployment model offers the greatest flexibility and ease of management. In both cases, the database will be started before the application starts. As with the ACFS file system dependency, the application should use the 'ora.database.db' , or ora.database.service.svc' Oracle Clusterware resource naming convention for the dependency definition. Please see the examples in the respective 'Syntax' sections.

AGCTL

AGCTL, Agent Control, is the agent command line utility to manage Oracle Grid Infrastructure agents (XAG) for application HA using Oracle Grid Infrastructure. AGCTL is the interface to add an application to the Oracle Grid Infrastructure as a clustered resource under agent control. Once configured, AGCTL is the management interface to bring the application online, offline or relocate, as well as to check the application state and modify the agent configuration. There are AGCTL options to disable and remove the application from the Oracle Grid Infrastructure agent control.

Installation and Configuration for Linux and UNIX

The Oracle Grid Infrastructure Agents require an operational installation of the Oracle Grid Infrastructure version 11.2.0.4 or later on all nodes where the application is targeted to run. The agents for 11.2.0.4 are available for download on http://oracle.com/goto/clusterware, on the "Downloads" tab. Oracle Grid Infrastructure 12*c* and later includes bundled agents as part of the standard distribution, installed by the Oracle Universal Installer in the Grid Infrastructure home directory. It is recommended to use the most recent download rather than the version bundled with the Oracle Grid Infrastructure distribution.

The agents and the AGCTL management interface are packaged for release 11.2.0.4 or later in the zip file xagpack.zip. To begin the installation, this zip file needs to be downloaded and expanded in a temporary directory. The $XAG_HOME and sub-directories must be owned by Oracle Grid Infrastructure install owner. The setup script, xagsetup.sh, must be run as the Oracle Grid Infrastructure install owner. The xagsetup.sh script offers the option to install local only (the default) or to install on all nodes in the cluster or, a subset of nodes in the cluster where the application is targeted to run. When deploying agents to multiple nodes in the cluster, the xagsetup.sh is executed on only one node of the cluster and the xagsetup.sh script will attempt to create $XAG_HOME directory on all remote nodes, with an identical absolute path and the correct ownership and permissions. Please review the readme.txt file in the $XAG_HOME directory for details.

An identical, operational installation of the application must be available on all nodes in the cluster targeted to support application failover. Application version, directory location, ownership and permissions must be the same on all nodes and the application must be configured conforming to the application documentation.

The bundled agents may be deployed as a new install where the agents are installed when the Oracle Grid Infrastructure is installed and configured. As many applications depend on application VIPs, the Oracle Grid Infrastructure should be configured and functional prior to installing the application. The Oracle Grid Infrastructure and (bundled) agent installs are orthogonal to an existing application deployment. This allows installation of the Oracle Grid Infrastructure and the bundled agents to be in separate directory structures and configuration of the Oracle Grid Infrastructure stack in an existing application environment. In this situation, typically the application stack will need to be stopped, the correct communication endpoint for the application to use will be configured for the respective VIP and the application will be restarted under agent control using the AGCTL interface.

```
$ ./xagsetup.sh –h

Setup Oracle Grid Infrastructure Agents agent to run in Oracle Clusterware

Usage: xagsetup.sh --install --directory <installdir>
       [--nodes <node>[,...] | --all_nodes]

Usage: xagsetup.sh --deinstall [--force]
       [--nodes <node>[,...] | --all_nodes]

Options:-

  --install                      Install Oracle Grid Infrastructure Agents
                                 to run in Oracle Clusterware
  --directory                    Installation directory
  --deinstall                    De-install Oracle Grid Infrastructure
                                 Agents from Oracle Clusterware
  --nodes <node>[,...]           Install/De-install Oracle Grid
                                 Infrastructure Agents on specified nodes
  --all_nodes                    Install/De-install Oracle Grid
                                 Infrastructure Agents on all nodes
  --force                        Remove all Oracle Grid Infrastructure
                                 Agents resources during de-install
```

Installation and Configuration for Windows

The Oracle Grid Infrastructure Agents require an operational installation of the Oracle Grid Infrastructure version 11.2.0.4 or later on all nodes where the application is targeted to run. The agents for 11.2.0.4 are available for download on http://oracle.com/goto/clusterware, on the "Downloads" tab. Oracle Grid Infrastructure 12*c* and later includes bundled agents as part of the standard distribution, installed by the Oracle Universal Installer in the Grid Infrastructure home directory.

The agents and the AGCTL management interface are packaged for release 11.2.0.4 in the zip file xagpack.zip. To begin the installation, this zip file needs to be downloaded and expanded in a temporary directory. The setup script, xagsetup.bat, must be run as the Oracle Grid Infrastructure install owner. The xagsetup.bat script offers the option to install local only (the default) or to install on all nodes in the cluster or, a subset of nodes in the cluster where the application is targeted to run. When deploying agents to multiple nodes in the cluster, the xagsetup.bat is executed on only one node of the cluster and the xagsetup.bat script will attempt to create $XAG_HOME directory on all remote nodes, with an identical absolute path and the correct ownership and permissions. Please review the readme.txt file in the $XAG_HOME directory for details.

An identical, operational installation of the application must be available on all nodes in the cluster targeted to support application failover. Application version, directory location, ownership and permissions must be the same on all nodes and the application must be configured conforming to the application documentation.

The bundled agents may be deployed as a new install where the agents are installed when the Oracle Grid Infrastructure is installed and configured. As many applications depend on application VIPs, the Oracle Grid Infrastructure should be configured and functional prior to installing the application. The Oracle Grid Infrastructure and (bundled) agent installs are orthogonal to an existing application deployment. This allows installation of the Oracle Grid Infrastructure and the bundled agents to be in separate directory structures and configuration of the Oracle Grid Infrastructure stack in an existing application environment. In this situation, typically the application stack will need to be stopped, the correct communication endpoint for the application to use will be configured for the respective VIP and the application will be restarted under agent control using the AGCTL interface.

```
c:\Windows\temp> xagsetup.bat -h

Setup Oracle Grid Infrastructure Agents agent to run in Oracle Clusterware

Usage: xagsetup.bat --install --directory <installdir>
       [--nodes <node>[,...] | --all_nodes]

Usage: xagsetup.bat --deinstall [--force]
       [--nodes <node>[,...] | --all_nodes]

Options:-

  --install                     Install Oracle Grid Infrastructure Agents
                                to run in Oracle Clusterware
  --directory                   Installation directory
  --deinstall                   De-install Oracle Grid Infrastructure
                                Agents from Oracle Clusterware
  --nodes <node>[,...]          Install/De-install Oracle Grid
                                Infrastructure Agents on specified nodes
  --all_nodes                   Install/De-install Oracle Grid
                                Infrastructure Agents on all nodes
  --force                       Remove all Oracle Grid Infrastructure
                                Agents resources during de-install
```

Requirements for Running AGCTL

AGCTL can be run as the Oracle Grid Infrastructure owner or a defined application administrator. The application administrator must be in the primary OS group of the Oracle Grid Infrastructure owner. For example, the Oracle Grid Infrastructure is installed by the oracle user in the group oinstall. The application administrator must also be in the oinstall group.

All AGCTL commands can be run by an application administrator. When creating application virtual IPs, administrators can either use the Oracle Grid Infrastructure utility appvipcfg, located in the Oracle Grid Infrastructure bin directory, or the VIP can be created using the AGCTL utility. In both cases, as the VIP is considered a super-user (root) owned resource, root access is required to create the VIP. If an application VIP is pre-created by the Oracle Grid Infrastructure administrator (as root), the application admin need only pass in the *--vip_name* on the AGCTL command line. This does not require super-user (root) privileges.

The Oracle Grid Infrastructure Agents and the AGCTL management interface support multiple application administrators for distinct applications. Application resource ownership and execution is controlled by UNIX like ACL definitions for a given resource and guarantees secure role separation. Role separation, execution privileges and ownership for an application are defined when the management of the application is presented to the Oracle Grid Infrastructure using the AGCTL management interface.

AGCTL Syntax and Use

```
$ ./agctl

Manages Apache Tomcat, Apache Webserver, E-Business Suite Concurrent Manager, GoldenGate, JDE
Enterprise Server, MySQL Server, Peoplesoft App Server, Peoplesoft Batch Server, Peoplesoft PIA Server,
Siebel Gateway, Siebel Server, WebLogic Administration Server as Oracle Clusterware Resources

 Usage: agctl <verb> <object> [<options>]

    verbs: add|check|config|disable|enable|modify|query|relocate|remove|start|status|stop

    objects:
apache_tomcat|apache_webserver|ebs_concurrent_manager|goldengate|jde_enterprise_server|mysql_server
|peoplesoft_app_server|peoplesoft_batch_server|peoplesoft_pia_server|siebel_gateway|siebel_server|webl
ogic_admin_server

 For detailed help on each verb and object and its options use:

    agctl <verb> --help or

    agctl <verb> <object> --help
```

AGCTL is the command to be used for application resource management. It is not supported to use the Oracle Grid Infrastructure utility CRSCTL to manage application resource of type xag, however, CRSCTL status calls for resource status, such as `` `*crsctl status resource –t*` ``, are supported.

XAG Upgrade

When XAG v9.1 is installed, the installer will automatically upgrade all existing XAG types & resources to the latest release (v9.1). Existing resources will be managed & monitored by the agents from the latest release. The prior XAG version installations can be deleted at any time after XAG v9.1 is installed.

In case for any reason, the automatic upgrade of XAG is not desirable and existing XAG resources should continue operating with the prior XAG versions, please run the installer with the *--no_auto_upgrade option* :-

```
$ ./xagsetup.sh --install --directory <...> --no_auto_upgrade
```

In some cases, the automatic upgrade may fail if some XAG resources or types were registered earlier as *'root'* user (for e.g if the 'agctl add ...' was run as *'root'* in the prior XAG version). In such cases, the XAG automatic upgrade will fail with the following error message :-

```
$ ./xagsetup.sh --install --directory <...> --all_nodes
```

```
CRS-0245:  User doesn't have enough privilege to perform the operation"
Please rerun 'agctl upgrade deployments' as 'root'.
```

If this error is hit, please run *'agctl upgrade deployments'* as *'root'* after the XAG installation is complete. This command will upgrade all existing XAG resources & types to the latest version (v9.1).

```
# /u01/xagv9/bin/agctl upgrade deployments
```

Oracle Grid Infrastructure Agents for Applications

Oracle GoldenGate

Oracle GoldenGate is a real-time database replication and data integration product. An Oracle GoldenGate instance consists of the ***manager*** process and its child processes, related programs, configuration files, checkpoint files, trail files and other logs/traces/templates/scripts. The child processes are named: *extract*, which captures change data, and *replicat*, which applies change data. There is exactly one running GoldenGate instance per installation of GoldenGate. A single ***manager*** process runs per GoldenGate instance/installation and it can support multiple ***extract*** and ***replicat*** processes.

Version Support Matrix

The following combination of Clusterware/GoldenGate releases is supported.

Grid Infrastructure (GI)	GoldenGate
11.2.0.4.+/12.1+/12.2+/18+/19+	11.1+/11.2+/12.1+/12.2+, 12.3+(Classic Edition)/18+/19+

*GI 11.2.0.4.+ release implies the same Oracle Automatic Storage Manager (ASM) release.

Agent Functions

- Manage GoldenGate application failover
- Start the GoldenGate instance manager process
- Monitor the GoldenGate instance extract process
- Monitor the GoldenGate instance replicat process
- Monitor the GoldenGate instance manager process
- Stop the GoldenGate instance and relevant dependencies
- Relocate the GoldenGate instance and relevant dependencies
- Clean the GoldenGate instance and relevant dependencies after a non-recoverable failure event

Resource Dependency Options

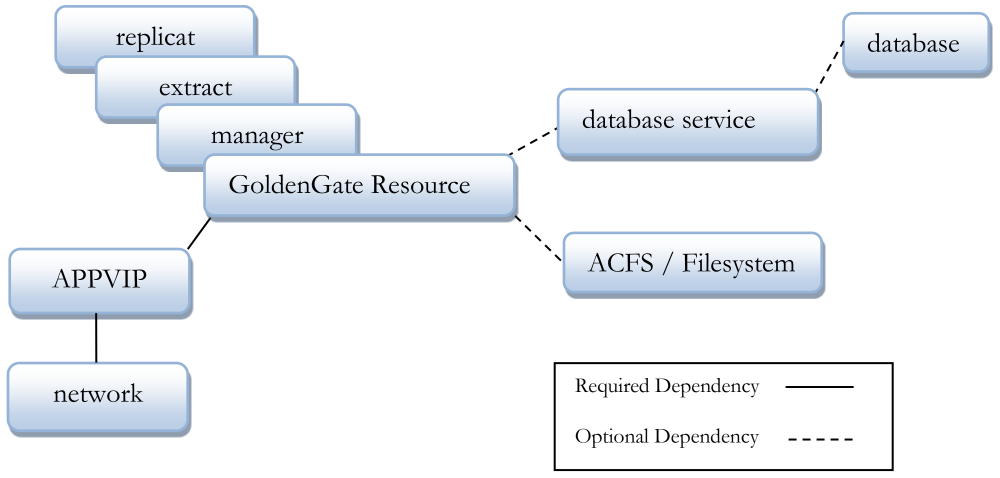


Figure 1

As noted in **Figure 1** above, the GoldenGate instance resource requires an application VIP (APPVIP in the example above) for manager connectivity by remote GoldenGate instances and for management and monitoring software components. An APPVIP requires a network resource net.net1.network, where net1 is the network number of the resource. Alternative network resources can be used if they exist. Other optional resource dependencies include ACFS or other supported file systems, and database connectivity dependencies either on the database service or the database directly.

State Definitions

ONLINE – The GoldenGate instance is online

OFFLINE – The GoldenGate instance is offline

INTERMEDIATE – The GoldenGate manager is online, however some or all extract and replicate processes are offline or have timed out when attempting to start

UNKNOWN - The state when Oracle Clusterware is unable to manage the resource and manual Oracle Clusterware intervention is required to stop it fix the root cause. Once corrected agctl start/stop commands should be used

The GoldenGate instance resource will transition to states: ONLINE, OFFLINE based on the operations and state of the *manager*. This resource will transition to state INTERMEDIATE if any of the specified *extract* or *replicat* processes are detected not to be running.

Resource Type Definitions

Oracle Clusterware uses resource types to organize and manage resources with common or similar attributes. The Oracle GoldenGate resource will have an application specific resource type, xag.goldengate.type, as defined internally by the Oracle Clusterware.

Monitoring of GoldenGate extract and replicat (ER) processes

The --monitor_extracts and --monitor_replicats parameters provide a basic level of monitoring of GoldenGate ER processes. If any extract or replicat process in the specified lists stops running, the GoldenGate resource transitions to INTERMEDIATE state.

GoldenGate ER processes are automatically restarted after a failure by the GG *manager,* based on the AUTORESTART parameter settings. Deployments should appropriately configure this parameter to make sure that ER processes are restarted when they abend. It is possible that an ER process runs into repeated failures on restart, thereby exhausting all restart attempts.

In such situations where an ER process is highly critical but not running anymore, a failover of the entire GoldenGate instance to another node in the cluster may solve the issue with the failed ER process. Deployments should configure the --critical_extracts and --critical_replicats with the list of ER processes that are critical and their repeated failures will cause the GoldenGate instance to failover to another node in the cluster.

Oracle DataGuard Integration

GoldenGate is commonly used in conjunction with Oracle DataGuard where Oracle GoldenGate replicates to remote sites, independent of the DataGuard standby. GoldenGate instances are configured on the primary site and standby site(s). Such deployments can use the *--dataguard_autostart* parameter to ensure that the GoldenGate instance is correctly started/stopped when a DataGuard planned/unplanned switchover takes place. As detailed in the GoldenGate documentation, *role based services* need to be used in the DataGuard setup, for example, a service with the PRIMARY role should be created on the primary and standby sites.

This service is then specified in the database connect string of the GoldenGate ER processes. When registering the GoldenGate resource on the primary and standby sites, the appropriate Clusterware resource name of the configured role based service needs to be passed to agctl using the --db_services parameter. When the *--dataguard_autostart* flag is set, a planned or unplanned DataGuard switchover ensures the running GG instance on the old primary will be stopped and the GG instance on the new primary will be automatically started.

If a Virtual IP resource is associated with the GG resource in a DataGuard setup, the *agctl add/modify goldengate* commands must be run as **root**. This ensures that the Virtual IP resource is also updated to automatically stop when the GG resource is stopped on a DataGuard role switchover. Thus the same IP address can be specified in the virtual IP resources on both the primary and standby sites. The GoldenGate administrative user & group must be explicitly given permissions to start and stop the GG resource:

```
# agctl add goldengate ggpri  --vip_name ggsourcevip --dataguard-autostart yes \
                              --user oracle --group oinstall
```

GoldenGate Agent AGCTL Syntax

Complete AGCTL usage for the GoldenGate resource is exposed using agctl –h. The following are common agctl operations for the GoldenGate application.

AGCTL command to register and configure a GoldenGate resource for a GoldenGate instance:

```
agctl [add | modify] goldengate instance_name
       --gg_home <GoldenGate installation directory>
       --serverpool <serverpool name> | --nodes node<,...>
       --instance_type <source|target|dual>
       --oracle_home <path>
       --db_services <associated database services>
        --use_local_services
       --databases <associated database resources>
       --environment_vars name=value1<,...>
       --monitor_extracts ext<,...>
       --monitor_replicats rep<,...>
       --critical_extracts  cext<,...>
       --critical_replicats  crep<,...>
       --jagent_autostart <yes|no>
       --network <network_number>
       --vip_name <VIP resource name>
       --ip <new VIP address>
        --user <user>
        --group <group>
       --filesystems acfs<,...>
       --attribute name=value<,...>
```

Where the options for *agctl add* and *modify* commands for GoldenGate are:

instance_name	The name of the GoldenGate instance/resource. The instance_name must be unique and cannot be changed after the resource is registered. **Required**
gg_home	The GoldenGate installation directory. **Required**
serverpool	The name of the server pool in which this GoldenGate instance should be started
nodes	A list of nodes where the GoldenGate instance can be run
instance_type	Indicates whether this is a source or target instance. For bi-directional replication both source and target are required definitions. This parameter is optional and is for informational purpose only.
oracle_home	The ORACLE_HOME location. **Required**
databases	The Oracle Database resources if GoldenGate instance has to be co-located with a Database instance and ORACLE_SID based connection is used by the extract/replicat processes. The ora.<database>.db resource name from `crsctl status resource –t` must be used for the databases definition.

db_services	The Database services (TNS), if the extract/replicat processes use service-based connection strings. The ora.<database>.<service_name>.svc name from `crsctl status resource –t` must be used for db_services definition.
use_local_services	This parameter (without any value) needs to be specified if the OGG instance must be co-located on the node where the required database service (TNS) is running. By default (when this option is not specified), the OGG instance and the database service can run on different nodes of the cluster.
monitor_extracts	An optional list of *extracts* to monitor. If any of the *extract* is not running, the GoldenGate instance resource state will transition to INTERMEDIATE.
monitor_replicats	An optional list of *replicats* to monitor. If any of the *replicat* is not running, the GoldenGate instance resource state will transition to INTERMEDIATE.
critical_extracts	An optional list of *critical* extracts to monitor. If any extract in the list has ABENDED and all local restart attempts of the extract have exhausted, the GoldenGate instance is failed over to another node.
critical_replicats	An optional list of *critical* replicats to monitor. If any replicat in the list has ABENDED and all local restart attempts of the replicat have exhausted, the GoldenGate instance is failed over to another node.
jagent_autostart	Setting this parameter to **yes** will ensure that the GoldenGate *jagent* process is started and stopped along with the GoldenGate instance. The appropriate configuration and licenses for the *jagent* component should be available.
dataguard_autostart	Set this parameter to **yes** to indicate that the GoldenGate instance is operating in a Oracle DataGuard setup and needs to be automatically started and stopped.
environment_vars	An optional list of environment variables to be passed when the GoldenGate instance is started/stopped/monitored. This is useful for setting up the environment when GoldenGate is operating on non-Oracle datastores.
vip_name	Specify an existing VIP resource (e.g created via *appvipcfg*) that is used by the OGG instance. This virtual-IP will always be active on the node where the OGG instance is running and can be used by remote clients/tools to connect to the OGG instance. If a new virtual-IP needs to be created for the OGG instance, the network/ip/user/group parameters can be specified (explained below)
network	The network number if a new VIP resource is to be created **Required if *agctl* needs to create the new VIP (virtual IP) resource. (see note below)**

ip	The VIP address if a new VIP resource is to be created **Required if *agctl* needs to create the new VIP (virtual IP) resource.**
user	The name of the OS user that owns the OGG instance (e.g oracle or ogg). **Required if *agctl* needs to create the new VIP (virtual IP) resource.**
group	The name of the administrative group that owns the OGG instance (e.g oinstall). **Required if *agctl* needs to create the new VIP (virtual IP) resource.**
filesystems	The filesystem resource dependency name for the GoldenGate instance. The dependency name must be the resource name known to the Oracle Clusterware, for example ora.data.ggacfs.acfs, where **data** is the ASM diskgroup and **ggacfs** is the volume.

Note: The application VIP can be created in advance by the Grid Admin as root using the *appvipcfg* command. If the application VIP is created in advance only the *--vip_name* needs to be defined in the *agctl add* command and the command can be executed as the GoldenGate admin. If creating the application VIP with *agctl*, the command must be run as root, the *--vip_name* should not be defined and the *--network*, *--ip --user* and *–group* parameters must be defined.

The following are examples of common *agctl* commands for GoldenGate.

This command lists the current configuration of the GoldenGate instance. This command will also print a brief descriptive message listing the resource attributes:

```
agctl config goldengate <instance_name>
```

When the resource is first registered, it is enabled by default. Otherwise, to enable the GoldenGate resource so that Clusterware can start the GoldenGate instance:

```
agctl enable goldengate <instance_name>
```

The following command disables the GoldenGate resource after which Clusterware cannot start the instance. This command can be used if the GoldenGate instance should not be run for a period of time:

```
agctl disable goldengate <instance_name>
```

The relocate command relocates the running GoldenGate instance from one serverpool to another or from one node to another:

```
agctl relocate goldengate instance_name
      [--serverpool serverpool_name | --node node_name]
```

To delete the GoldenGate resource from the Clusterware registry:

```
agctl remove goldengate instance_name [--force]
```

This command starts the GoldenGate instance on the specified node or serverpool:

```
agctl start goldengate instance_name
     [--serverpool serverpool_name | --node node_name]
```

To get the current known state of the GoldenGate instance:

```
agctl status goldengate instance_name [-node node_name]
```

To stop the running GoldenGate instance:

```
agctl stop goldengate instance_name
```

Sample Configuration

The following is an example of a GoldenGate extract and replicat configuration on 2 4-node clusters where the Oracle Grid Infrastructure is installed and configured for both clusters. A GoldenGate instance containing 2 extracts is registered with Clusterware using agctl, as root (Windows: Administrator)

For Linux:

```
# agctl add goldengate gg_1 --gg_home /myacfs/ogg  --instance_type source \
--nodes host1, host2, host3, host4 \
--network 1 --ip 10.10.10.10 --user oracle  --group oinstall \
--filesystems ora.data.ggacfs.acfs  --databases ora.orcl1.db \
--oracle_home /u01/app/oracle/product/11.2.0/dbhome_1 --monitor_extracts etest,ptest
```

For Windows:

```
c:\> agctl add goldengate gg_1 --gg_home c:\myacfs\ogg  --instance_type source --nodes host1,
host2, host3, host4 --network 1 --ip 10.10.10.10 --user oracle  --group ora_install
--filesystems ora.data.ggacfs.acfs  --databases ora.orcl1.db
--oracle_home c:\app\oracle\product\12.1.0.2.0\dbhome_1 --monitor_extracts etest,ptest
```

Where:

-the GoldenGate instance is gg_1
-the GoldenGate home is /myacfs/ogg (windows: c:\myacfs\ogg)
-the GoldenGate instance type is *source* for extract
-the nodes in the cluster participating in GoldenGate HA are
host1
host2
host3
host4
-the network is the default ora.net1.network and the VIP is 10.10.10.10
-the GoldenGate user is oracle in the group oinstall
-the file system dependencies are on the ACFS resource ora.data.ggacfs.acfs as defined by the grid admin
-the database dependency is on the database resource ora.orcl1.db
-the database home is /u01/app/oracle/product/11.2.0./dbhome_1
(windows: c:\app\oracle\product\12.1.0.2.0\dbhome_1)
-and the extracts to monitor are etest for primary change data capture and ptest for sending trail files to remote systems.

On the replication target cluster, the GoldenGate instance with a single replicat is registered using *agctl*, as root:

```
# agctl add goldengate gg_2 --gg_home /myacfs/ogg  --instance_type target \
--nodes host5,host6,ost7,host8 \
--network 1 --ip 10.10.10.20 --user oracle  --group oinstall \
--filesystems ora.data.ggacfs.acfs  --databases ora.orcl2.db\
--oracle_home /u01/app/oracle/product/11.2.0/dbhome_1 --monitor_replicats rtest
```

Where:

-the GoldenGate instance is gg_2
-the GoldenGate home is /myacfs/ogg
-the GoldenGate instance type is *target* for replicat
-the participating nodes are
host5
host6
host7
host8
-the network is the default ora.net1.network and the VIP is 10.10.10.20
-the GoldenGate user is oracle in the group oinstall
-the file system dependencies are on the acfs resource ora.data.ggacfs.acfs as defined by the grid admin
-the database dependency is on the database ora.orcl2.db
-the database home is /u01/app/oracle/product/11.2.0./dbhome_1
-and the replicat to monitor is *rtest*.

Oracle GoldenGate Agent Notes

1. Extracts with remote trail file writes must use the target VIP in the RMTHOST setting.
2. Control of the GoldenGate application will be through *agctl start/stop/status/relocate* commands. Do not stop the manager process via the GoldenGate command line interface GGSCI or via the GoldenGate GUI products as this will initiate a component failover.
3. If there are multiple GoldenGate instances corresponding to separate GoldenGate installation paths, a Clusterware GoldenGate resource needs to be registered for each clustered instance.
4. Oracle Clusterware will start, stop, monitor, restart, and failover the GoldenGate ***manager*** process. Clusterware will not start, stop or restart individual *extract* and *replicat* processes.
5. The Oracle Grid Infrastructure Agent for GoldenGate leverages the GoldenGate native AUTOSTART and AUTORESTART parameters in the *manager* parameter file for ***extract*** and ***replicat*** HA. Therefore, these parameters must be setup appropriately so that the ***extract*** and ***replicat*** processes are started and kept running. Clusterware can optionally monitor specified ***extract*** or ***replicat*** processes if configured to do so (see add command line syntax above) but their status will not initiate a failover in the current version.
6. After GoldenGate integration with Oracle Clusterware, only the agctl tool should be used to start, stop and relocate the GoldenGate instance. Clusterware will implement pre-execution checks and ensure dependent infrastructure resources like databases, filesystems, etc. are running before the GoldenGate processes are started.

NOTE: *agctl* is run as root in the configuration example because the application vip (APPVIP) is created on the *agctl* command line and not pre-created using *appvipcfg*. When creating the APPVIP using *agctl*, you must define the user who will manage the VIP.

GoldenGate administrators must ensure the required ***extracts*** and ***replicats*** (or both) are configured for AUTOSTART and AUTORESTART. If the GoldenGate resource transitions to INTERMEDIATE state, Administrators need to take corrective actions for the failed ***extract(s)/replicat(s)*** and then restart them.

Oracle GoldenGate Integrated Capture is fully supported. Refer to Oracle Support MOS note 1557031.1 – ***Oracle GoldenGate – Oracle RDBMS Server Recommended Patches***. If using a version 11.2.0.4 database, apply the specific bundle patch for Integrated Extract 11.2.x (see Oracle Support MOS note, Doc ID 1411356.1).

For more detail on Oracle GoldenGate and bundled agent deployments please see the "Oracle GoldenGate Best Practices: Configuring Oracle GoldenGate with Oracle Grid Infrastructure Agents (XAG)".

Oracle GoldenGate 12.3 (Microservices Architecture)

Oracle GoldenGate Microservices Architecture is a new administration architecture that provides REST-enabled services as part of the Oracle GoldenGate environment. The REST-enabled services provide remote configuration, administration, and monitoring through HTML5 web pages, command line, and APIs.

Version Support Matrix

Grid Infrastructure (GI)	GoldenGate
12.1+/12.2+/18+/19+	12.3.0.1.3+/18+/10+

agctl Options Specific to GoldenGate Microservices Architecture

service_manager	Passing this parameter (no additional value required) indicates that the OGG instance is Microservices Architecture type.
config_home	OGG deployment configuration home directory (OGG_CONF_HOME)
var_home	OGG deployment variable home directory (OGG_VAR_HOME)
Port	Service Manager port number
Adminuser	OGG Microservices Administrator account
ogg_container	The name of the Oracle GoldenGate Docker container. Passing this parameter indicates that the OGG is on a Docker container.

agctl will prompt for the Administrator account password and store the entered password in a secure wallet. These credentials will be used to login to the Service Manager and perform monitoring and startup tasks.

The following parameters **do not apply** to OGG Microservices Architecture instances: *instance_type, monitor_extracts, monitor_replicats, dataguard_autostart, critical_extracts, critical_replicates, jagent_autostart.*

The following parameters are common to both types of OGG instances and their descriptions are available in the previous section: *gg_home, oracle_home, databases, db_services, filesystems, serverpools, nodes, environment_vars, ip, network, user*

Example for registering a GoldenGate Microservices Architecture instance

```
# agctl add goldengate ggsca1 --gg_home /u01/ogg12.3   \
--service_manager
–config_home /mnt/acfs_gg/goldengate/deployments/ggsm01/etc/conf
--var_home /mnt/acfs_gg/goldengate/deployments/ggsm01/var
--port 9100 –adminuser admin
--nodes host5,host6,host7,host8 \
--network 1 --ip 10.10.10.20 --user oracle  --group oinstall \
--filesystems ora.datac1.acfs_gg.acfs  --databases ora.orcl2.db\
--oracle_home /u01/app/oracle/product/11.2.0/dbhome_1
```

Example for registering a GoldenGate Microservices Architecture on Docker container

```
# agctl add goldengate ggsca1 --gg_home /u01/ogg12.3   \
--service_manager
--ogg_container ogg
--config_home /mnt/acfs_gg/goldengate/deployments/ggsm01/etc/conf
--var_home /mnt/acfs_gg/goldengate/deployments/ggsm01/var
--port 9100 –adminuser admin
--nodes host5,host6,host7,host8 \
--network 1 --ip 10.10.10.20 --user oracle  --group oinstall \
--filesystems ora.datac1.acfs_gg.acfs  --databases ora.orcl2.db\
--oracle_home /u01/app/oracle/product/11.2.0/dbhome_1
```

All other *agctl* commands like *start/stop/relocate/modify/status/check/config/remove/enable/disable* have same parameters and same semantics as detailed in the previous section.

Example of *agctl config* output for a GoldenGate Microservices Architecture instance:-

```
# agctl config goldengate ggsca1
Instance name: ggsca1
Application GoldenGate location is: /u01/ogg12.3
Goldengate MicroServices Architecture environment : yes
OGG Container: ogg
Goldengate Service Manager configuration directory : mnt/acfs_gg/goldengate/deployments/ggsm01/etc/conf
Goldengate Service Manager var directory : /mnt/acfs_gg/goldengate/deployments/ggsm01/var
Service Manager Port : 9100
Goldengate Administration User : admin
```

The OGG Adminstrator user password can be modified by running the *agctl modify* command. Below example shows how to modify the Service Manager port and set a new password for the OGG Administrator account.

```
# agctl modify goldengate ggsca1 –port 9200 –adminuser admin
Enter password for 'admin' :
```

Oracle GoldenGate Microservices Agent Notes

1. The *agctl add* command needs to be run as the 'root' user when registering the OGG instance. Pass the –user & --group parameters to specify the OS user (e.g oracle or ogg etc) and group (e.g oinstall) owning the OGG instance and having privileges to run the OGG instance. The OGG instance will be started by Oracle Clusterware under the specified user account.

2. The *agctl modify* commands needs to be run as the 'root' user if the OGG Administrator user or its password are being changed.

3. Oracle Clusterware will start the ServiceManager process and monitor the health of the OGG instance. The ServiceManager is responsible for starting other OGG processes like *adminserver, recvserver, distserver, extract, replicat etc.*

4. Oracle Grid Infrastructure version 12.1.0.2 or higher is required to register the GoldenGate Microservices instance.

Oracle Siebel Customer Relationship Management (CRM)

Oracle Siebel CRM is a suite of Customer Relationship Management applications of which the Siebel Gateway Server, Siebel Servers and the Siebel Web Server Extensions are critical server components for application availability. Each of these components maintain essential configuration, logging and application metadata. The Grid Infrastructure Bundle Agents for Oracle Siebel CRM provide high availability for the Siebel Gateway Server and configured Siebel Servers.

Version Support Matrix

The Oracle Grid Infrastructure for Siebel CRM supports the following combination of Oracle Grid Infrastructure / Oracle Siebel releases. The respective Database versions must be compatible with the Grid Infrastructure and Siebel versions.

Grid Infrastructure	Siebel
11.2.0.4.+/12.1+/12.2+/18+/19+	8.1+

Oracle Siebel CRM Supported Deployment Models

The Oracle Grid Infrastructure Agents for Siebel CRM High Availability support the following three deployment models depicted in **Figure 2**:

- Model 1: Standalone Singleton Siebel Gateway Server with failover
- Model 2: Standalone Siebel Server with failover
- Model 3: Siebel Gateway Server with Siebel Server with failover

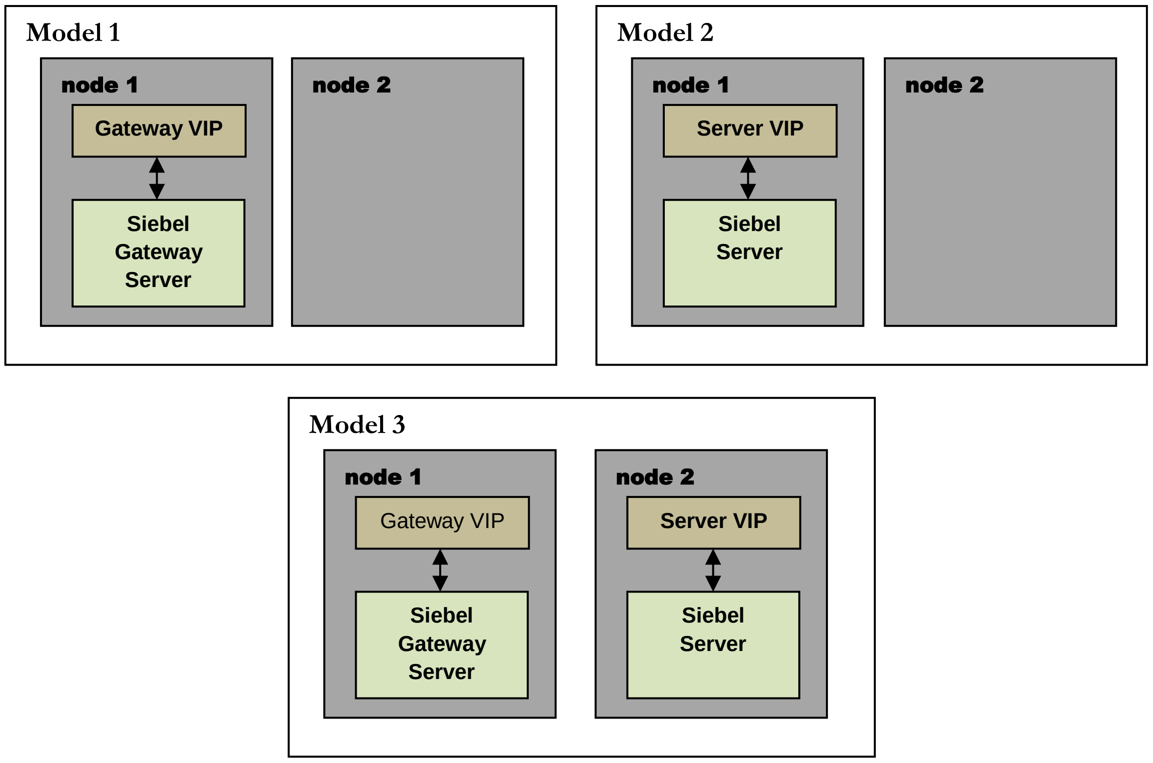


Figure 2

These deployment models will permit co-location of the Siebel Gateway Server with the Siebel Server on the same physical server. And, if configured, each Siebel Server must be bound to a unique application Siebel Server VIP to preserve the Siebel Server HA model.

Siebel Gateway Server

The Siebel Gateway Name Server is a required singleton server process which coordinates access of Siebel Enterprise Server and Siebel Servers to configuration data and connectivity information. When the Siebel Gateway Name Server is down, no new Siebel Server components can be started or added and server administration functions are severely limited.

Siebel Server

The Siebel Server is the middle-tier platform that supports both back-end and interactive processes for every Siebel client. These processes are components within the Siebel Server architecture and support functions such as:

- Mobile Web client synchronization
- Operation of business logic for Siebel Web clients, as well as connectivity and access to the Siebel Database and Siebel File System
- Integration with legacy or third-party data
- Automatic assignment of new accounts, opportunities, service requests, and other records
- Workflow management

The Siebel Server supports both multi-process and multithreaded components, and can operate components in background, batch, and interactive modes.

Agent Functions

- Manage the Siebel application failover for the Gateway and Siebel Servers
- Start the Siebel Gateway Server or Siebel Server and relevant dependencies
- Monitor the Siebel Gateway Server or Siebel Server and relevant dependencies
- Stop the Siebel Gateway Server or Siebel Server and relevant dependencies
- Relocate the Siebel Gateway Server or Siebel Server and relevant dependencies
- Clean the Siebel Gateway Server or Siebel Server and relevant dependencies after a non-recoverable failure event

Resource Dependency Options

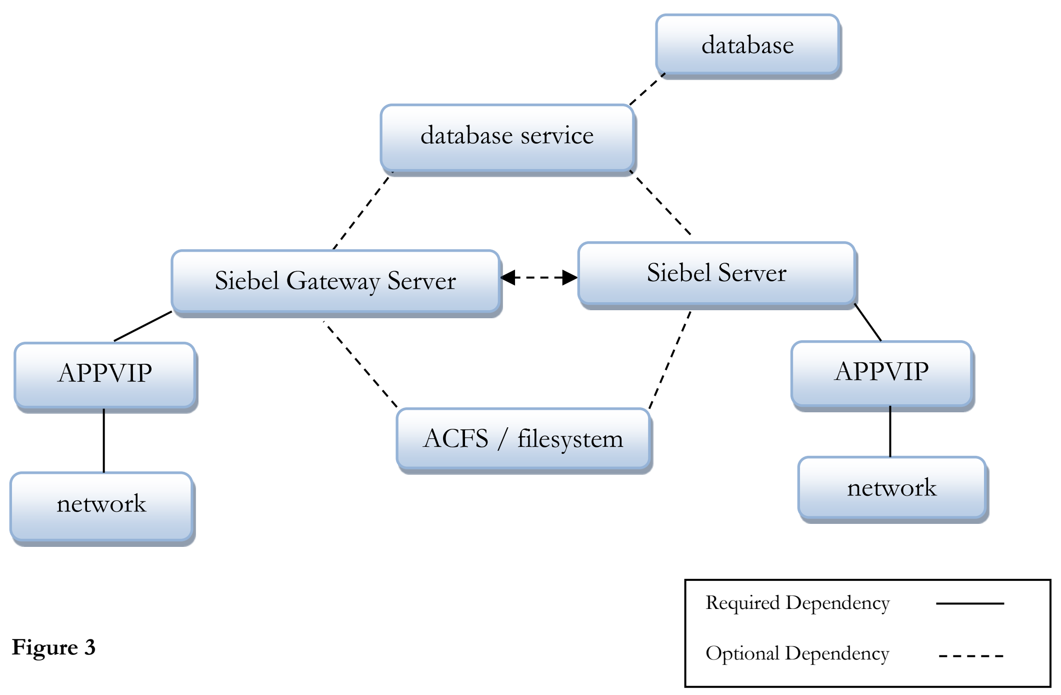


Figure 3

The Siebel Gateway Server, as depicted in **Figure 3**, requires an application VIP (APPVIP) and supporting network resource. The Siebel Server and any additional configured Siebel Servers each require a unique application VIP to preserve individual Siebel Server HA. All application VIPs may share the same network resource (the net1 network resource by default) or they can configure unique network resource dependencies on unique network adapters. Optional start and stop dependencies can be configured for the Siebel Gateway Server and configured Siebel Servers where starting one of these resources will start the other. In addition, start and stop dependencies may be configured for a supporting shared file system such as ACFS. The agent supports start and stop dependencies configured for either the database or configured services depending on how the application connects to the database.

State Definitions

ONLINE – The Siebel Gateway Server or Siebel Server is online

OFFLINE – The Siebel Gateway Server or Siebel Server is offline

INTERMEDIATE – The Siebel Gateway Server or Siebel Server is online, however inaccessible

UNKNOWN - The state when Oracle Clusterware is unable to manage the resource and manual Oracle Clusterware intervention is required to stop it fix the root cause. Once corrected agctl start/stop commands should be used

Resource Type Definitions

Oracle Clusterware uses resource types to organize and manage resources with common or similar attributes. The Oracle Siebel CRM resource will have an application specific resource type, xag.siebgtwy.type for the Siebel Gateway Server and xag.siebsrvr.type for the Siebel Server, as defined internally by the Oracle Clusterware.

Siebel Agent AGCTL Syntax

The *agctl add* command adds Siebel Gateway Server or Siebel Server instance to Oracle Clusterware. The Siebel user can add the Siebel Gateway Server or Siebel Server and all of its options when adding the resources with pre-created VIP., The options *network_number, ip_address,* must be executed as root in order to create the VIP and the user and group options must be defined.
The following are common *agctl* commands for the Siebel agent.

AGCTL command to register and configure a Siebel Gateway Server resource for a Gateway Server instance:

```
agctl [add | modify] siebel_gateway <instance_name>
         --siebel_home <siebel_home>
         --oracle_home <oracle_home>
         --oracle_client_home <oracle_client_home>
         --serverpool <serverpool> | --nodes <node1[,…]>
         --vip_name <vip_resource> | --network=<network_number>
                                     --ip=<ip_address>
                                     --user=<user> --group=<group>
         --attribute "<attribute_name=value,attribute_name=value,…>"
         --filesystems <filesystem1[,…]>
         --databases <database1[,…]>
         --db_services <db_service1[,…]>
         --environment_vars "<name1=value1[,…]>"
```

AGCTL command to register and configure a Siebel Server resource for a Siebel Server instance:

```
agctl [add | modify] siebel_server <instance_name>
         --siebel_home <siebel_home>
         --oracle_home <oracle_home>
         --enterprise_name <enterprise_name>
         --server_name < server_name>
         --oracle_client_home <oracle_client_home>
         --gateway_resource <gateway_resource> ]
         --serverpool <serverpool> | --nodes <node1[,…]>
         --vip_name <vip_resource> | --network=<network_number>
                                     --ip=<ip_address>
                                     --user=<user> --group=<group>
         --attribute "<attribute_name=value,attribute_name=value,…>"
         --filesystems <filesystem1[,…]>
         --databases <database1[,…]>
         --db_services <db_service1[,…]>
         --environment_vars "<name1=value1[,…]>"
```

Where the options for *agctl add* and *modify* commands for Siebel are:

instance_name	The name of the Siebel Gateway instance/resource. The *instance_name* must be unique and cannot be changed after the resource is registered. **Required**
siebel_home	The Siebel Gateway installation directory. **Required**
oracle_client_home	The location of the Oracle 32-bit Client Libraries for client connectivity to the database
enterprise_name	The Enterprise Server Name **Required**
serverpool	The name of the server pool in which this Siebel instance/resource should be started
nodes	A list of nodes where the Siebel Gateway instance/resource can be run
environment_vars	An optional list of environment variables to be passed when the Siebel Gateway or Server instance is started/stopped/monitored.
network	The network number if a new VIP resource is to be created **Required**
ip	The VIP address if a new VIP resource is to be created **Required** if not using pre-created VIP
user	The name of the OS Siebel user. **Required** if not using pre-configured VIP
group	The name of the Siebel group to which the Siebel user belongs.
attribute	Sets/overrides default values for standard Clusterware attributes of the Siebel resource (e.g. CHECK_INTERVAL, AUTO_START)
filesystems	List of file system resource dependencies for this Siebel instance. The filesystem resource name must be the resource name known to the Oracle Clusterware, for example ora.<diskgroup>.<volume>.acfs, for ACFS filesystems.
databases	Oracle Database resource dependency for the Siebel instance. The ora.<database>.db resource name from \`crsctl status resource –t\` must be used for the **databases** definition.
db_services	Database services dependency name. The ora.<database_name>.<service_name>.svc resource name from \`crsctl stat resource –t\` must be used for the **db_services** definition.

The following are further examples of common *agctl* commands for Siebel:

To list the current configuration of the requested Siebel instance and print a brief descriptive message listing the resource attributes:

```
agctl config { siebel_gateway | siebel_server } [ <instance_name> ]
```

This command enables the requested Siebel resource so that Clusterware can start the instance. When the resource is first registered, it is enabled by default:

```
agctl enable { siebel_gateway | siebel_server } <instance_name> [ --node
                <node_name> ]
```

This command disables the requested Siebel instance after which Clusterware cannot start the instance. This command can be used if the required Siebel instance should not be run for a period of time:

```
agctl disable { siebel_gateway | siebel_server } <instance_name> [ --node
<node_name> ]
```

To relocate the running Siebel instance (*siebel_gateway* or *siebel_server*) from one serverpool to another or from one node to another:

```
agctl relocate { siebel_gateway | siebel_server } <instance_name>
                    [ --server <serverpool> | --node <node_name> ]
```

This command deletes the requested Siebel resource. The force option will be required if there are running dependencies on the resource:

```
agctl remove { siebel_gateway | siebel_server } <instance_name> [ --force ]
```

To start the requested Siebel instance on the specified node or serverpool:

```
agctl start { siebel_gateway | siebel_server } <instance_name>
                    [ --server <serverpool> | --node <node_name> ]
```

This command returns the current known state of the requested Siebel resource:

```
agctl status { Siebel_gateway | Siebel_server } <instance_name> [-node
<node name> ]
```

This command stops the running Siebel resource where *instance_name* is either the Siebel Gateway Server or the Siebel Server instance. The force flag may be required if there are running dependencies on the requested resource:

```
agctl stop { siebel_gateway | siebel_server } <instance_name> [ --force ]
```

Sample Configurations

The following is an example of a Siebel Gateway Server HA configuration on a 2-node cluster where the Oracle Grid Infrastructure is installed and configured prior to the Siebel Gateway Server and the VIP and the VIP hostname is used as the Siebel Gateway address rather than the physical hostname.

To create the Siebel Gateway Server execute as root:

```
# agctl add siebel_gateway sieb_gtwy --siebel_home /myacfs/gtwy  \
--oracle_home /u01/app/oracle/product/11.2.0/dbhome_1 \
--oracle_client_home /u01/app/oracle/product/11.2.0/client
--nodes ost144-41,ost144-42 \
--network 1 --ip 10.10.121.141 --user siebel  --group oinstall \
--filesystems ora.data.sieb.acfs  --databases ora.orcl4.db
```

Where:
- -the Siebel Gateway Server instance is sieb_gtwy
- -the Siebel Gateway Server home is /myacfs/gtwy
- -the Oracle Home is /u01/app/oracle/product/11.2.0/dbhome_1
- -the Oracle Client Library home is /u01/app/oracle/product/11.2.0/client
- -the nodes in the cluster participating in Siebel Gateway Server HA are
    - ost144-41
    - ost144-42
- -the network used is the default ora.net1.network and the VIP is 10.10.121.141
- -the Siebel user is siebel in the group oinstall
- -the file system dependency is on the resource ora.data.sieb.acfs as defined by the grid admin
- -the database dependency is on the database resource ora.orcl4.db

To create the Siebel Server HA for a single server, execute as root:

```
# agctl add siebel_server  sieb_srvr1  --siebel_home /myacfs/sieb_srvr1  \
--oracle_home /u01/app/oracle/product/11.2.0/dbhome_1 \
--client_home /u01/app/oracle/product/11.2.0/client \
--nodes ost144-41,ost144-42 \
--network 1 --ip  10.10.121.142 --user  siebel  --group  oinstall \
--filesystems  ora.data.sieb.acfs  --databases  ora.orcl4.db
```

Where:
- -the Siebel Server instance is sieb_srvr1
- -the Siebel Server home is /myacfs/sieb_srvr1
- -the Oracle Home is /u01/app/oracle/product/11.2.0/dbhome_1
- -the Enterprise Server name is SIEB
- -the Oracle Client Library home is /u01/app/oracle/product/11.2.0/client
- -the nodes in the cluster participating in Siebel Server HA are
    - ost144-41
    - ost144-42
- -the network used is the default ora.net1.network and the VIP is 10.10.121.142
- -the Siebel user is siebel in the group oinstall

-the file system dependency is on the resource ora.data.sieb.acfs as defined by the grid admin
-the database dependency is on the database ora.orcl4.db

Siebel CRM Agent Notes

There are three deployment models that influence the Siebel Server to Siebel Gateway communication. For new installs, configure the application VIP and use that as the Siebel Gateway Name Server host and IP for the Siebel Gateway and Siebel Server installations. If you are installing the Oracle Grid Infrastructure Agents in a pre-existing Siebel environment you will need to execute the following steps:

1. Create the Siebel Gateway VIP on the *agctl* command line or using *appvipcfg*
2. Stop the Siebel Gateway Server
3. Execute the `*agctl add siebel_gateway …*` per instructions above where *--vip_name or --network, --ip, --user* are defined.
4. Start the Siebel Gateway instance using `*agctl start siebel_gateway <siebel_instance_name>*`

When step 3 is executed, the agctl utility will modify the Siebel Servers dependent on that Gateway Server by executing, internal to the `*agctl add siebel_gateway*`, the following steps:

a) source siebenv.sh
b) Stop Siebel Server service (stop_server ALL)
c) Remove *.osdf in the $SIEBEL_ROOT/sys
d) Backup the SVC file with the format svc.siebsrvr.siebel:<Siebel Server>
e) Remove the svc.* files in the $SIEBEL_ROOT/sys
f) Recreate SVC file by executing siebctl command below to bind the VIP to the Siebel Servers:

```
$ siebctl -S siebsrvr -i $enterprise_name:$server_name -a -g "-g $ip:$gateway_port -e $enterprise_name
-s $server_name” (executed by agctl)
```

Where:
enterprise_name = The name of the Siebel Enterprise
siebsrvrname = The name of the Siebel Server
ip:gtwyport = The agent VIP assigned to the Siebel Gateway Server:port number
server_name= The name of the Siebel Server

This ensures that the clustered Siebel Servers associated with the defined Siebel Gateway are updated automatically. Whenever the *agctl add* or *modify* Siebel_gateway command is run and the *--vip_name* or *--network, --ip, --user* have been modified, agctl will run the above steps a-f automatically so that the Siebel Servers recognize the Siebel Gateway Server’s new network identity. The changes take effect when Siebel Servers are started. For any Siebel Servers added subsequent to adding or modifying the Siebel Gateway address, the user will need to execute the steps a-f noted above manually.

Siebel Server Changes to Enable Siebel Enterprise Server Communication

When the Siebel Server is clustered you need to ensure that other non-clustered Siebel Servers in the Siebel Enterprise can communicate with the clustered Siebel Server. The *agctl* command will not modify non-clustered

Siebel Servers. Please follow the next set of steps to manually change the hostname of the clustered Siebel Server to the Cluster Network Name:

1. Using the srvrmgr command in order to bind to the Siebel Server cluster hostname to the Siebel Server change the configuration of Siebel server using srvrmgr command below:
   a. If the Siebel version is 7.7 or later, change the ServerHostAddress parameter to the IP address of the Siebel server's logical host name resource using the following syntax:

```
$ srvrmgr> change param ServerHostAddress=$ip for server $server_name
```

   b. If the Siebel version is earlier than 7.7, change the Host parameter to the logical hostname for the Siebel server using the following syntax:

```
$ srvrmgr:hasiebel> change param Host=$ip for server $server_name
```

NOTE: The above can either be IP or Hostname as long as they map to the cluster's logical IP/logical Hostname**.** The changes take effect when Siebel Servers are started.
Control of the Siebel components must be through *agctl start/stop/status/relocate* commands. Do not stop the Siebel Gateway or Siebel Server processes via the Siebel command line interface or via any other means as this will initiate a component failover.

Apache HTTP and Tomcat Server

The Oracle Grid Infrastructure Agents manage Apache Tomcat and HTTP Servers for high availability and scalability. The bundled agents provide HA for these Apache components which is unavailable by default. The ability to configure and control multiple Apache resources provides a highly scalable framework which administrators can manage through a simple, intuitive interface.

Apache Tomcat

Apache Tomcat is an open source software implementation of the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed under the Java Community Process. Apache Tomcat, Tomcat, Apache, the Apache feather, and the Apache Tomcat project logo are trademarks of the Apache Software Foundation (see tomcat.apache.org)

Apache HTTP Server

Apache HTTP Server is a public-domain open source Web server (see httpd.apache.org).

Version Support Matrix

The Oracle Grid Infrastructure for Apache Tomcat and HTTP Servers supports the following combination of Oracle Grid Infrastructure / Apache releases.

Supported Apache Versions

Grid Infrastructure (GI)	HTTP Server	Tomcat Server
11.2.0.4+/12.1+/12.2+/18+/19+	2.2.x, 2.0.x	8.0.x, 7.0.x, 6.0.x, 5.5.x

Apache Tomcat and Web Server Supported Deployment Models

Oracle Grid Infrastructure bundled agents for Apache supports four deployment models depicted in **Figure 4** below. The first model (model 1) is a standalone Apache HTTP Server. The second model (model 2) is a standalone Apache Tomcat Server without the HTTP Server. The third model (model 3) is the Apache HTTP Server co-located with the Apache Tomcat Server on a single node in the cluster. The fourth model (model 4) is the HTTP Server and the Apache Tomcat Server deployed on different nodes in the cluster.

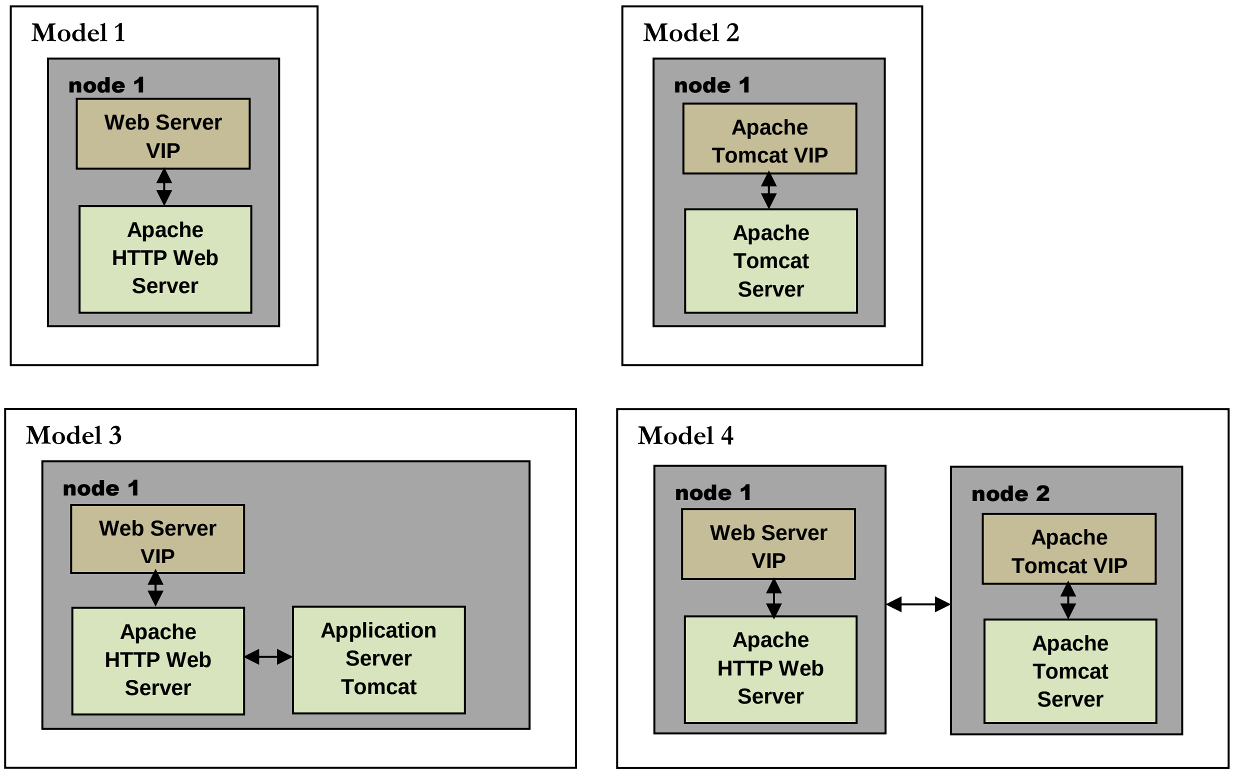
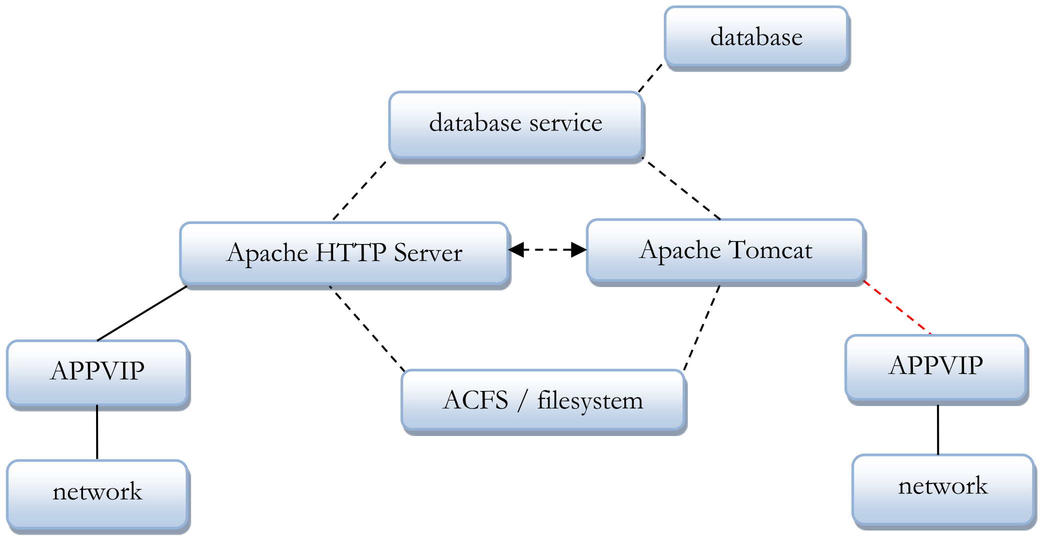


Figure 4

Apache Agent Functions

- Manage the Apache Tomcat and and/or Apache HTTP Server failover
- Manage the Apache Tomcat and/or Apache HTTP Server scalability
- Start the Apache Tomcat and/or Apache HTTP Server and relevant dependencies
- Monitor the Apache Tomcat and/or Apache HTTP Server and relevant dependencies
- Stop the Apache Tomcat and/or Apache HTTP Server and relevant dependencies
- Relocate the Apache Tomcat and/or Apache HTTP Server and relevant dependencies
- Clean the Apache Tomcat and/or Apache HTTP Server and relevant dependencies after a non-recoverable failure event

Resource Dependency Options



Required Dependency	———
Optional Dependency	- - - - -
Optional or required based on deployment model	- - - - -

Figure 5

The chosen deployment model will dictate required dependencies. **Figure 5** illustrates that Models 1-3 have a single network resource stack defined for either the Apache HTTP Server or Tomcat Server. If the Apache HTTP Server and the Tomcat Server are co-located on a single node, the Apache HTTP Server will have a dependency on the network stack and the Apache Tomcat Server will have a dependency on the Apache HTTP Server. In deployment Model 4, the Apache HTTP Server and Apache Tomcat are running on separate nodes in the cluster. In this model, they would both need network stack (network, VIP) dependencies. How the Apache Tomcat Server is deployed will determine whether there is a required dependency on the network stack (see **red** dependency link above). Supplementary dependencies may be defined between the HTTP server and Tomcat even though they are running on separate nodes.

Apache HTTP Server Dependency Options

If the *--databases*, *--db_services,* and/or *--filesystems* option are specified, the Apache Web Server will have a start dependency on databases, database service, and file system resources. This dependency requires that the database, database service, and file system resources must be running before the Apache HTTP Server can start. The Apache HTTP Server can start on any node in the cluster as long as the database, database service, and file system resources are running. The agent dependency model guarantees that these resources will be started in the correct order, if configured. The resource dependency definition must be the resource names as exposed in \`crsctl stat res –t\`. For example database, database service and filesystem (for ACFS) would be represented as ora.<database_name>.db, ora.<database_name>.<service_name>.svc and ora.<diskgroup>.<volume>.acfs respectively.

Apache Tomcat Dependency Options

If the *--webserver*, *--databases*, *--db_services,* and/or *--filesystems* option is specified, the Apache Tomcat Server will have a start dependency on the Apache HTTP Server, database, database service, and file system resources. This dependency means that the Apache HTTP Server, database, database service, and file system resources must be running before the Apache Tomcat Server can start. The Apache Tomcat Server can start on any defined node in the cluster as long as the Apache HTTP Server, database, or database service is running on any node in the cluster. The agent dependency model guarantees these resources will be started in the correct order, if the dependency is configured.

Apache Tomcat will also have a start dependency on the Apache HTTP Server, database, database service, and/or file system. This dependency means that Apache Tomcat Server will automatically start whenever the Apache HTTP Server, database, database Service, and file system resources are started. As with the Apache HTTP Server, the resource dependency definition must be the resource names as exposed in `crsctl stat res –t`. For example database, database service and filesystem (for ACFS) would be represented as ora.<database_name>.db, ora.<database_name>.<service_name>.svc and ora.<diskgroup>.<volume>.acfs respectively.

Apache Agent State Definitions

ONLINE – The Apache HTTP Server or Apache Tomcat is online

OFFLINE – The Apache HTTP Server or Apache Tomcat is offline

INTERMEDIATE – The Apache HTTP Server or Apache Tomcat is online, however inaccessible

UNKNOWN - The state when Oracle Clusterware is unable to manage the resource and manual Oracle Clusterware intervention is required to stop it fix the root cause. Once corrected agctl start/stop commands should be used.

Resource Type Definitions

Oracle Clusterware uses resource types to organize and manage resources with common or similar attributes. The Apache HTTP Server and Apache Tomcat resources will have application specific resource type, xag.httpd.type and xag.tomcat.type as defined and used internally by the Oracle Clusterware.

Apache Agent AGCTL Syntax

Complete AGCTL usage for the Apache resources is exposed using agctl –h. The following are common agctl operations for the Apache HTTP Server and Apache Tomcat.

AGCTL command to register and configure an Apache HTTP Server resource:

```
agctl [add | modify] apache_webserver <instance_name>
         --apache_home <webserver_home>
         --config_file <configuration_file>
         --serverpool <serverpool> | --nodes <node1[,…]>
         --vip_name <vip_resource> | --network=<network_number>
                                     --ip=<ip_address>
                                     --user=<user> --group=<group>
         --attribute "<attribute_name=value,attribute_name=value,…>"
         --filesystems <filesystem1[,…]>
         --databases <database1[,…]>
         --db_services <db_service1[,…]>
         --environment_vars "<name1=value1[,…]>"
```

AGCTL to register and configure an Apache Tomcat resource for a Apache Tomcat instance:

```
agctl [add | modify] apache_tomcat <instance_name>
         --catalina_home <catalina_home>
         --java_home <java_home>
         --catalina_base <catalina_base>
         --jre_home <jre_home>
         --webserver <webserver_instance>
         --serverpool <serverpool> | --nodes <node1,node2[,…]>
         --vip_name <vip_resource> | --network=<network_number>
                                     --ip=<ip_address>
                                     --user=<user> --group=<group>
         --attribute "<attribute_name=value,attribute_name=value,…>"
         --filesystems <filesystem1[,…]>
         --databases <database1[,…]>
         --db_services <db_service1[,…]>
         --environment_vars "<name1=value1[,…]>"
```

Where the options for *agctl add* and *modify* commands for Apache HTTP and Tomcat Server are:

instance_name	Name of the Apache HTTP server or Tomcat server instance/resource. The instance_name must be unique and cannot be changed after the resource is registered. **Required**
apache_home	Web Server installed directory. **Required**
config_file	Name of the configuration file. It's used for multiple instances.
catalina_home	Application Server installed directory **Required**
catalina_base	Application Server configuration directory. It's used for multiple instances.

java_home	JAVA Development installed directory for Application Server. **Required**
jre_home	JRE Development installed directory for Application Server. Defaults to java_home if it's empty.
webserver	Name of the Web Server instance. This option is only used for adding Application Server(s) for model 3 & 4 only. Otherwise, it returns an error.
serverpool	Name of the server pool in which this Web Serve instance/resource should be started
environment_vars	An optional list of environment variables to be passed when the Apache HTTP or Tomcat instances are started/stopped/monitored.
vip_name	The virtual IP address of the node or the application VIP resource that was pre-created by Oracle Clusterware Application VIP utility (*appvipcfg*).
network	The network number if a new VIP resource is to be created **Required**
ip	The VIP address if a new VIP resource is to be created **Required** if not using pre-created VIP
user	The name of the OS Apache user. **Required** if not using pre-configured VIP
group	The name of the Apache OS group to which the Apache user belongs.
attribute	Set/override default values for standard Clusterware attributes of the Apache resource (e.g. CHECK_INTERVAL, AUTO_START)
filesystems	List of file system resource dependencies for this Apache instance. The filesystem resource name must be the resource name known to the Oracle Clusterware, for example ora.<diskgroup>.<volume>.acfs, for ACFS filesystems.
databases	Oracle Database resource Apache instance dependency. The database resource name must be the resource name known to the Oracle Clusterware, for example ora.<database_name>.db.
db_services	Database services resource Apache dependency. The db_services resource name must be the resource name known to the Oracle Clusterware, for example ora.<database_name>.<service_name>.svc.

The following are other common Apache *agctl* commands.

This command lists the current configuration of the requested Apache instance. This command will also print a brief descriptive message listing the resource attributes:

```
agctl config { apache_webserver | apache_tomcat } <instance_name>
```

This command enables the requested Apache resource so that agent can start the instance. When the resource is first registered, it is enabled by default:

```
agctl enable { apache_webserver | apache_tomcat } <instance_name> [--node
                <node_name>]
```

This command disables the requested Apache instance after which agent cannot start the instance. This command can be used if the required Apache instance should not be run for a period of time:

```
agctl disable { apache_webserver | apache_tomcat } <instance_name> [--node
                <node_name>]
```

This command relocates the running Apache HTTP or Tomcat Server instance from one serverpool to another or from one node to another:

```
agctl relocate { apache_webserver | apache_tomcat } <instance_name>
                    [--server <serverpool> | --node <node_name>]
```

This command deletes the requested Apache resource from the Clusterware registry. The force option will be required if there are running dependencies on the resource. This command does not remove the apache_webserver or application server defined homes:

```
agctl remove { apache_webserver | apache_tomcat } <instance_name> [ --force ]
```

This command starts the requested Apache instance on the specified node or serverpool:

```
agctl start { webserver | tomcat } <instance_name>
                    [--server <serverpool> | --node <node_name>]
```

This command returns the current known state of the requested Apache resource:

```
agctl status [ webserver | tomcat ] <instance_name> [--node <node_name>]
```

This command stops the running Apache resource where instance_name is either the Apache webserver or the Apache tomcat instance. The force flag may be required if there are running dependencies on the requested resource:

```
agctl stop { webserver | tomcat } <instance_name> [ --force ]
```

Sample Configurations

The following is an example of an Apache HA configuration on a 2-node cluster where the Apache HTTP Server is configured to run in a defined serverpool and the database connection if via TNS services:

```
# agctl add apache_webserver webserver01 --apache_home /apache2.2 \
--configuration_file /apache2.2/myconf.xml --network 1 --ip 192.168.56.201 \
--user apache --group oinstall --filesystems ora.data.apache_ws.acfs \
--server_pool ora.pool1 --db_services ora.orcl5.dbserv1.svc
```

Where:

-the Apache HTTP Server instance is webserver01
-the Apache HTTP Server home is /apache2.2
-the Apache configuration file location is /apache2.2/myconf.xml
-the network used is the default ora.net1.network and the VIP is 192.168.56.201
-the Apache user is apache in the group oinstall
-the file system dependency is on the resource ora.data.apache_ws.acfs as defined by the Grid Admin
-the serverpool is ora.pool1 (it is expected that the Grid Admin creates the serverpool in advance)
-the database dependency is on the database TNS service ora.orcl5.dbserv1.svc

The following command adds the Apache Tomcat Server and restricts the application to run on specific nodes in the cluster with a dependency on the Apache HTTP Server defined above where the Apache HTTP Server and the Apache Tomcat Server are not co-located. Note that in the previous command, the Apache HTTP Server VIP was created as root on the *agctl* command line with the defined *--user* option. Below, the Apache Tomcat VIP has been pre-created. This command is run as the Apache Tomcat Admin:

```
$ agctl add apache_tomcat tomcat01  --catalina_home /tomcat7.8 --java_home \
/usr --catalina_base /tomcat7_base --webserver webserver01 --vip_name myvip \
--nodes node2,node4 --filesystems ora.data.apache_tc.acfs
```

Where:

-the Apache Tomcat Server instance is tomcat01
-the Apache Tomcat Server Catalina home is /tomcat7.8
-the Jave home is /usr
-the Apache Tomcat Server Catalina base is /tomcat7_base
-the webserver dependency is on Apache HTTP Server instance webserver01
-the APPVIP is pre-configured and is myvip
-the Apache Tomcat Server instance is restricted to run on nodes node2 and node4
-the file system dependency is on the resource ora.data.apache_tc.acfs as defined by the Grid Admin

Apache Agent Notes

For Apache HTTP Server:

1. During *agctl add* or *modify* operations where --*vip_name* option is added or modified or --*network, --ip, --user* options are added or modified, *agctl* will modify the *Listen* entry in $APACHE_HOME/conf/httpd.conf *w*ith the the new VIP address.
2. If $APACHE_HOME location is on a shared file system, *agctl* will modify the configuration file for all nodes.
3. If $APACHE_HOME location is not on a shared file system, a warning will be issued. The Apache HTTP admin is responsible for updating the *Listen* entry in $APACHE_HOME/conf/httpd.conf file on every node to use the VIP address.

   For example:

   ```
   Listen <host>:80
   Will be modified to:
   Listen <ip address>:80
   Where <ip address> is the IP address of the application VIP
   ```

   NOTE: Prior to any modification of the *httpd.conf* file, the original file will will be saved as *httpd.conf.xag.*
   If *vip_name* is modified, the *httpd.conf* file will also be modified to reflect the changes.
   If *vip_name* is removed or if Web Server instance is removed from Oracle Clusterware, the *httpd.conf* file will be restored from the backup file *httpd.conf.xag.* The backup file will be deleted if restore is successful.

For Apache Tomcat:

1. During *agctl add* or *modify* operations where the --*vip_name* option is added or modified or --*network, --ip,* or –*user* options are added or modified, *agctl* will internally modify the <*Host name*= …> entry in $CATALINA_BASE/conf/server.xml file with the new VIP address.
2. If $CATALINA_BASE location is on a shared file system, *agctl* will automatically back up the server.xml file to server.xml.xag and modify the configuration file for all nodes.
3. If $CATALINA_BASE location is not on a shared file system, a warning will be issued. The Apache Tomcat admin is responsible for updating the *Listen* entry in $ CATALINA_BASE/conf/server.xml file on every node to use the new VIP address.

   For example:

   ```
   Host name=<host>:80
     must be modified to:
   Host name=<ip address>:80
     Where <ip address> is the IP address of the application VIP
   ```

   NOTE: Prior to any modification of the *server.xml* file by *agctl*, the original file will be saved as *server.xml.xag.* If the VIP resource is modified by changing--*vip_name* or, --*network*, --*ip* , --*user*, the *server.xml* file must also be manually modified to reflect the changes. If the VIP resource is removed my removing --*vip_name* or --*network, --ip, --user* or if the Apache Tomcat Server instance is removed using *agctl*, the server.xml the will be automatically restored from the backup file server.xml.xag. The backup file will be deleted if the restore is successful.

Control of the Apache components must be through *agctl start/stop/status/relocate* commands. Do not stop the Apache HTTP or Tomcat Server processes via the Apache command line interface or via any other means as this will initiate the component failover.

Oracle PeopleSoft Pure Internet Architecture

PeopleSoft applications are composed of the following main server elements: Web Browsers, Web Server(s), Application Server(s), Batch Server(s) (Process Scheduler), and Database Server. The Grid Infrastructure Bundled and Standalone agents version 3, (XAGv3), for Oracle PeopleSoft Applications provides HA and resource management functionality for the PeopleSoft Application Server(s), the PeopleSoft Batch Server(s) (Process Scheduler) and the PeopleSoft PIA Server(s). The WebLogic Server, in this release, is not a component of the Oracle Grid Infrastructure agent solution for PeopleSoft Servers.

Version Support Matrix

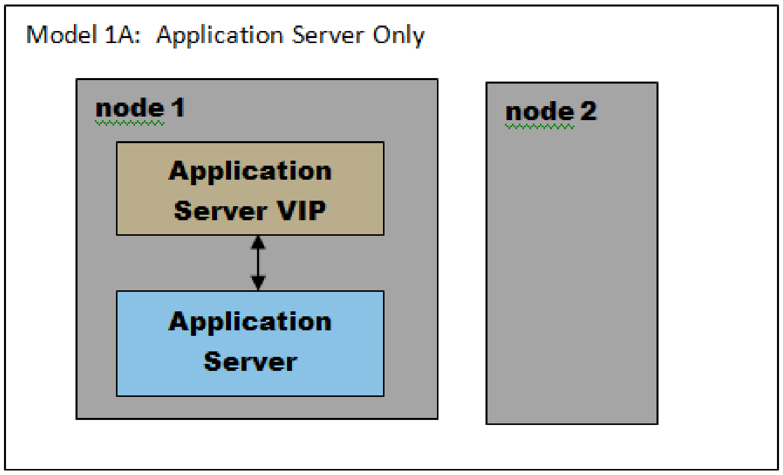
The Oracle Grid Infrastructure for PeopleSoft Applications supports the following combination of Oracle Grid Infrastructure / PeopleSoft CRM releases. The respective Database versions must be compatible with the Grid Infrastructure and PeopleSoft versions.

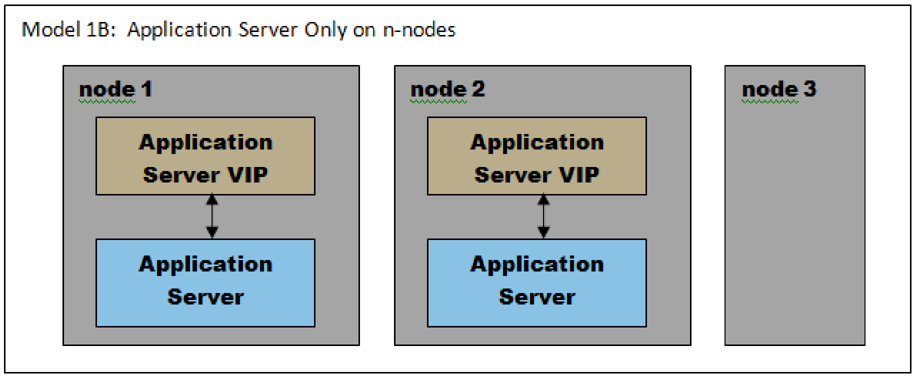
Grid Infrastructure	PeopleSoft
11.2.0.4.+/12.1+/12.2+/18+/19+	8.52+

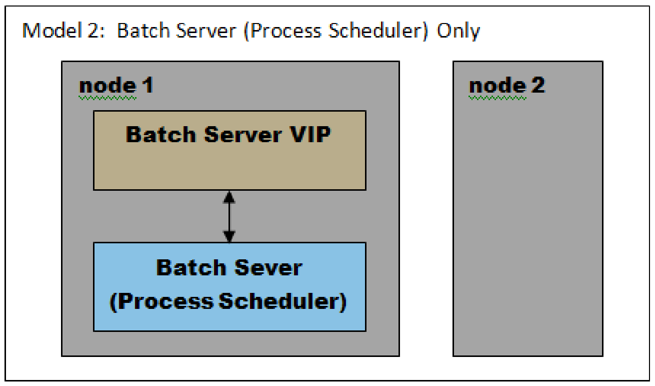
Oracle PeopleSoft Supported Deployment Models

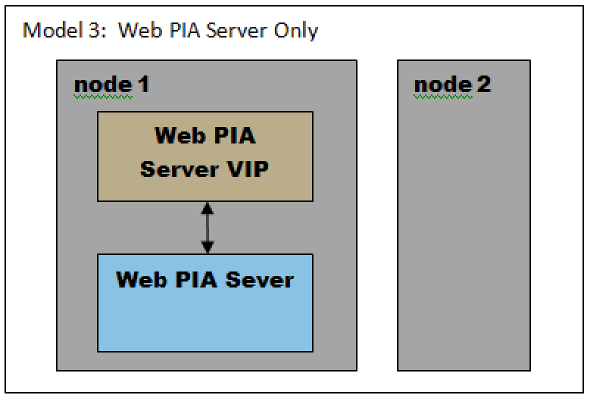
The Oracle Grid Infrastructure Agents for PeopleSoft CRM High Availability support the following 5 deployment models 1-4, depicted in **Figure 6:**

- Model 1A - Application Server(s) only Active/Standby
- Model 1B - Application Server(s) only Active/Active on *N* number of nodes
- Model 2 - Batch Server(s) (Process Scheduler) only Active/Standby
- Model 3 - PIA Server(s) only Active/Standby
- Model 4 - Application Server(s), Batch Server(s), and PIA Server(s)




Model 1B: Application Server Only on n-nodes
node 1
Application Server VIP
Application Server
node 2
Application Server VIP
Application Server
node 3


Model 2: Batch Server (Process Scheduler) Only
node 1
Batch Server VIP
Batch Sever (Process Scheduler)
node 2


Model 3: Web PIA Server Only
node 1
Web PIA Server VIP
Web PIA Sever
node 2

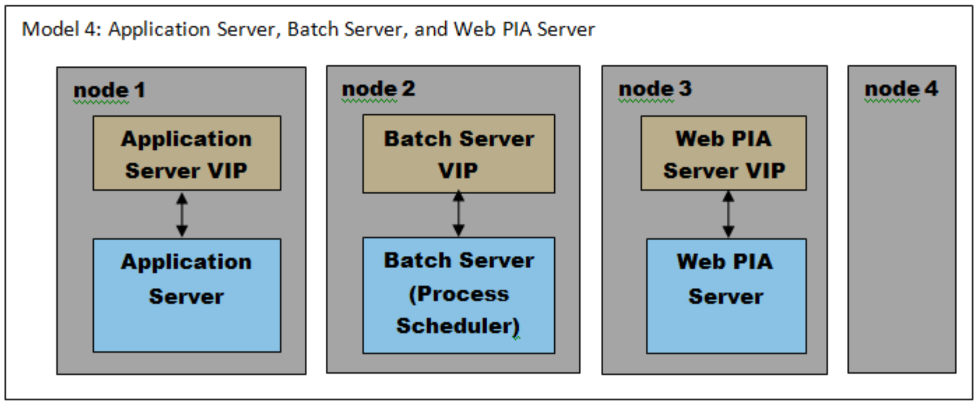

Model 4: Application Server, Batch Server, and Web PIA Server
node 1
Application Server VIP
Application Server
node 2
Batch Server VIP
Batch Server (Process Scheduler)
node 3
Web PIA Server VIP
Web PIA Server
node 4

Figure 6

These deployment models configure PeopleSoft Servers on distinct physical servers either as singleton, active/standby services as in the case of the Batch Server (Process Scheduler) and PIA Server, or, as possible for the Application Servers, multiple, active/active services running per Application Server domain. The Batch Server process can be configured on the same server as the database, using the agent interface to integrate with Oracle Clusterware. As described, the stack can be configured together across a PeopleSoft HCM cluster. Each PeopleSoft Server must be bound to a unique application VIP to preserve the high availability model.

PeopleSoft Application Server

The application server is the core of PeopleSoft Internet Architecture. It runs business logic and submits SQL to the database server. An application server consists of numerous PeopleSoft server processes, grouped in domains. Each server process within a domain provides unique processing abilities, enabling the application server to respond effectively to a multitude of transaction requests generated throughout the PeopleSoft architecture.

Application servers require database connectivity software installed locally to maintain the SQL connection with the RDBMS. You must install the required connectivity software and associated utilities for your RDBMS on any server on which you intend to run the PeopleSoft Application Server.

After the application server establishes a connection to the database, any device that initiates a transaction request through the application server takes advantage of the application server's direct connection to the database.[3]

PeopleSoft Batch Server (Process Scheduler)

The PeopleSoft Process Scheduler environment can also be thought of as the "batch" environment. It is the location where many of your batch programs, such as Application Engine programs, run, and in most situations, this is also where you have COBOL and SQR executables installed.

In a multi-server environment, you can decide where your Process Scheduler environment resides based on available servers and performance requirements. In the PeopleSoft topology, you can install Process Scheduler on a separate server, or it can run on either the application server or the database server.[4]

PeopleSoft PIA Server

A Java-enabled web server is required to extend the PeopleSoft architecture to the internet and intranet. When you install the PeopleSoft Internet Architecture on the web server, you install a collection of PeopleSoft Java servlets designed to handle a wide range of PeopleSoft transactions originating from the internet or an intranet. The web server connects to the Application Server which talks to the database server.[5]

Agent Functions

- Manage the PeopleSoft component failover for the PeopleSoft Application Servers, Batch Servers (Process Schedulers) and PIA Servers.
- Start the PeopleSoft components listed above and relevant dependencies
- Monitor the PeopleSoft components listed above and relevant dependencies

[3] Oracle http://docs.oracle.com/cd/E38689_01/pt853pbr0/eng/pt/tsvt/task_ApplicationServers-d27b3e.html
[4] Oracle http://docs.oracle.com/cd/E38689_01/pt853pbr0/eng/pt/tsvt/task_PeopleSoftProcessSchedulerServer-896bf0.html
[5] http://docs.oracle.com/cd/E38689_01/pt853pbr0/eng/pt/tsvt/task_WebServer-89682a.html

- Stop the PeopleSoft components listed above and relevant dependencies
- Relocate the PeopleSoft components listed above and relevant dependencies
- Clean the PeopleSoft components listed above and relevant dependencies after a non-recoverable failure event

Resource Dependency Options

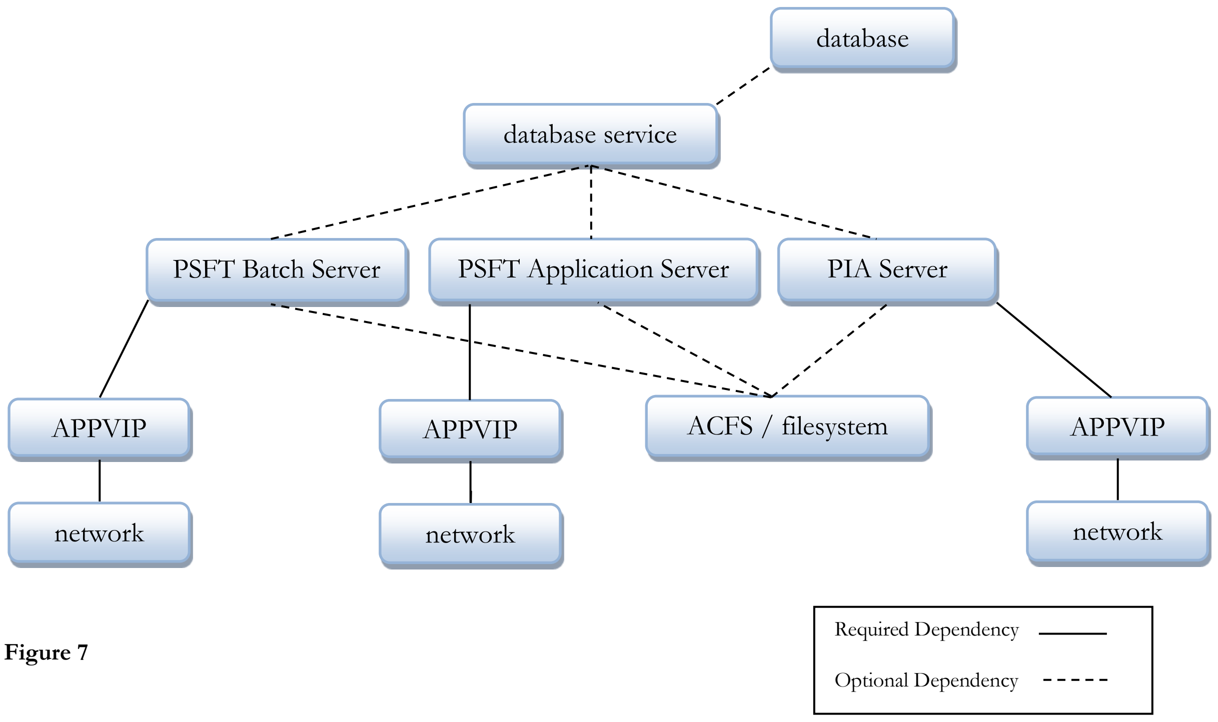


Figure 7

As indicated in Figure 7 above, each PeopleSoft Batch Server, Application Server and PIA Server instance resource requires a unique application VIP (shown as APPVIP) for the manager connectivity by remote client applications and for both management and monitoring of the server components. An APPVIP requires a network resource, e.g., net.1.network, where 1 is the network id configured for this Grid deployment. Other optional resource dependencies include ACFS or other supported file systems, and database connectivity dependencies either on the database service or the database directly.

PeopleSoft Application Server, Batch Server, PIA Server, Database, Database Service, and File System Dependencies

The following constraints will be used to establish the dependencies between Application Server, Batch Server, PIA Server, Database, Database Service, and File system.

PeopleSoft Application Server Start / Stop Dependency

If the *databases, db_service,* and/or *filesystems* option is specified, the Application Server will have a *hard(global:database,db_service,filesystem)* start dependency on Database, Database Service, and File System resource. A *hard* dependency means Database, Database Service, and File System resource must be running before Application Server can start. The *global* modifier means Application Server can start as long as the Database, Database Service, and file system resource are running on any node of the cluster.

PeopleSoft Batch Server Start / Stop Dependency

If the *databases, db_service,* and/or *filesystems* option is specified, the Batch Server will have a *hard(global:database,db_service,filesystem)* start dependency on Database, Database Service, and file system resource. A *hard* dependency means Database, Database Service, and file system resource must be running before Batch Server can start. The *global* modifier means Batch Server can start as long as Database, and Database Service are running on any node of the cluster.

PeopleSoft PIA Server Start / Stop Dependency

If the *databases, db_service,* and/or *filesystems* option is specified, the PIA Server will have a *hard(global:database,db_service,filesystem)* start dependency on Database, Database Service, and file system resource. A *hard* dependency means Database, Database Service, and file system resource must be running before Batch Server can start. The *global* modifier means PIA Server can start as long as Database, and Database Service are running on any node of the cluster.

State Definitions

ONLINE – The PeopleSoft component is online

OFFLINE – The PeopleSoft component is offline

INTERMEDIATE – The PeopleSoft component is online, however inaccessible

UNKNOWN - The state when Oracle Clusterware is unable to manage the resource and manual Oracle Clusterware intervention is required to stop the resource and fix the root cause. Once corrected *agctl* start/stop commands should be used to control the component.

Resource Type Definitions

Oracle Clusterware uses resource types to organize and manage resources with common or similar attributes. The Oracle PeopleSoft component resources will have the following application specific resource types defined internally by the Oracle Clusterware:

The PeopleSoft Application Server resources will be of the xag.psoftapp.type type.
The PeopleSoft Batch Server resources will be of the xag.psoftbatch.type type.
The PeopleSoft PIA Server resources will be of the xag.psoftpia.type type.

PeopleSoft Agent AGCTL Syntax

The *agctl add* command adds People Server instances to Oracle Clusterware. The PeopleSoft admin can add the PeopleSoft components and all of its options when adding the resources with pre-created VIP. The options *network_number, ip_address,* must be executed as root in order to create the VIP and the user and group options must be correctly defined.

The following are common *agctl* commands for the PeopleSoft agent. These commands illustrate the syntax and options when adding the PeopleSoft Appliction Server, the PeopleSoft Batch Server and the PeopleSoft PIA Server components to the Oracle Clusterware for agent control:

AGCTL command to register and configure a PeopleSoft Application Server resource for an Application Server instance:

```
agctl [add | modify] peoplesoft_app_server <instance_name>
         --ps_home <ps_home>
         --ps_cfg_home <ps_cfg_home>
         --tuxdir <tuxedo_directory>
         --server_domain <server_domain>
         --serverpool <serverpool> | --nodes <node1[,…]>
         --vip_name <vip_resource> | --network=<network_number>
                                     --ip=<ip_address>
                                     --user=<user> --group=<group>
         --attribute "<attribute_name=value,attribute_name=value,…>"
         --filesystems <filesystem1[,…]>
         --databases <database1[,…]>
         --db_services <db_service1[,…]>
         --environment_vars "<name1=value1[,…]>"
```

AGCTL command to register and configure a PeopleSoft Batch Server resource for a Batch Server instance:

```
agctl [add | modify] peoplesoft_batch_server <instance_name>
         --ps_home <ps_home>
         --ps_cfg_home <ps_cfg_home>
         --tuxdir <tuxedo_directory>
         --scheduler_database <scheduler_database>
         --serverpool <serverpool> | --nodes <node1[,…]>
         --vip_name <vip_resource> | --network=<network_number>
                                     --ip=<ip_address>
                                     --user=<user> --group=<group>
         --attribute "<attribute_name=value,attribute_name=value,…>"
         --filesystems <filesystem1[,…]>
         --databases <database1[,…]>
         --db_services <db_service1[,…]>
         --environment_vars "<name1=value1[,…]>"
```

AGCTL command to register and configure a PeopleSoft PIA Server resource for a PIA Server instance:

```
agctl [add | modify] peoplesoft_pia_server <instance_name>
       --ps_home <ps_home>
       --ps_cfg_home <ps_cfg_home>
       --tuxdir <tuxedo_directory>
       --server_domain <server_domain>
       --pia_home <pia_home>]
       --serverpool <serverpool> | --nodes <node1[,…]>
       --vip_name <vip_resource> | --network=<network_number>
                                   --ip=<ip_address>
                                   --user=<user> --group=<group>
       --attribute "<attribute_name=value,attribute_name=value,…>"
       --filesystems <filesystem1[,…]>
       --databases <database1[,…]>
       --db_services <db_service1[,…]>
       --environment_vars "<name1=value1[,…]>"
```

Where the options for *agctl add* and *modify* commands for PeopleSoft components are:

instance_name	Name of the Application Server, Batch Server, or PIA Server instance. The *instance_name* must be unique and cannot be changed after the resource is registered. **Required**
ps_home	The path of the PeopleSoft installed installed directory. **Required**
ps_cfg_home	The path of the PeopleSoft domain configuration files. **Required**
pia_home	The location of the PeopleSoft Pure Internet Architecture domains. Note: If pia_home is specified, it overrides the use of ps_cfg_home.
tuxdir	The path of the Tuxedo installation directory. **Required**
server_domain	The name of the PeopleSoft Server Domain. Note: The option is applied to Application Server or PIA Server. **Required**
scheduler_database	The name of the database that's associated with a Process Scheduler Server Agent. Note: The option is applied only to Batch Server. **Required**
serverpool	Name of the server pool that the Application Server, Batch Server, or PIA Server instance can be run.
nodes	List of nodes where Application Server, Batch Server, or PIA Server instance can be run.
vip_resource	An optional list of environment variables to be passed when the Application Server, Batch Server, or PIA Server instance is started/stopped/monitored.
network	The network number if a new VIP resource is to be created **Required**

ip	The VIP address if a new VIP resource is to be created **Required** if not using pre-created VIP
user	The name of the PeopleSoft OS user who owns the application VIP. **Required** if not using pre-configured VIP *The user who is adding the VIP resource and the owner of VIP resource are not necessary the same. For example, 'root' user can add the VIP resource, but its owner is foo user.*
group	The name of the PeopleSoft OS group to which the PeopleSoft user belongs.
attribute	Sets/overrides default values for any attribute that can be applied to the Application Server, Batch Server, or PIA Server resource. For example: *-attribute "CHECK_INTERVAL=60,FAILURE_INTERVAL=30"* *Note: the time is measure in seconds*
filesystems	List of file system resource(s) that need to be mounted before the Application Server, Batch Server, or PIA Server can start. These would usually be the file system(s) containing the application home/binaries. *Note: The filesystems and db_services are mutually exclusive.*
databases	List of database instances created for Application Server, Batch Server, or PIA Server instance.
db_services	List of data base services(s) that need to be up and running before Application Server, Batch Server, or PIA Server can start. Note: The *filesystems* and *db_services* are mutually exclusive.
environment_vars	An optional list of environment variables to be passed when the Application Server, Batch Server or PIA Server instances are started/stopped/monitored.

The following are further examples of common *agctl* commands for the PeopleSoft components:

config

The *agctl config* command displays Application Server, Batch Server, or PIA Server configuration store in Oracle Clusterware.

Note: The *instance_name* is optional. If the *instance_name* is not specified, all configured instances will be displayed.

```
agctl config { peoplesoft_app_server | peoplesoft_batch_server |
  peoplesoft_pia_server }
```

enable

The *agctl enable* command enables the Application Server, Batch Server, or PIA Server instance so that it can run under Oracle Clusterware for automatic startup, failover, or restart. The Oracle Clusterware application supporting Application Server, Batch Server, or PIA Server instance may be up or down to use this function. The default value is *enabled*.

If the Application Server, Batch Server, or PIA Server instance is already enabled, then the command is ignored. Enabled objects can be started, and disabled objects cannot be started:

```
agctl enable { peoplesoft_app_server | peoplesoft_batch_server |
  peoplesoft_pia_server } <instance_name> [ --node <node_name> ]
```

disable

The *agctl disable* command disables the Application Server, Batch Server, or PIA Server instance. Use the *agctl disable* command when Application Server, Batch Server, or PIA Server instance must shut down for maintenance. The disabled object does not automatically restart.

```
agctl disable { peoplesoft_app_server | peoplesoft_batch_server |
  peoplesoft_pia_server } <instance_name> [ --node <node_name> ]
```

modify

The *agctl modify* command modifies the configuration for Application Server, Batch Server, or PIA Server. The PeopleSoft user can modify the Application Server, Batch Server, or PIA Server and all of its options, except for the *network_number, ip_address,* and *user* options. If these options are specified, it must be run as root user.

relocate

The *agctl relocate* command relocates Application Server, Batch Server, or PIA Server instance from one node to another node or from one server pool to another serverpool.

```
agctl relocate { peoplesoft_app_server | peoplesoft_batch_server |
  peoplesoft_pia_server} <instance_name> [ --server_pool <serverpool> | --node
```

remove

The *agctl remove* command removes the Application Server, Batch Server, or PIA Server configuration information from Oracle Clusterware. Environment settings for the object are also removed.

Notes: This command does not destroy the Application Server, Batch Server, or PIA Server installed root directory and does not remove from its domain.

Removing the Application Server or Batch Server is allowed only if it has no dependency, unless the *--force* option is used.

```
agctl remove { peoplesoft_app_server | peoplesoft_batch_server |
  peoplesoft_pia_server } <instance_name> [ --force ]
```

start

The *agctl start* command starts the Application Server, Batch Server, or PIA Server instance in Oracle Clusterware.

```
agctl start { peoplesoft_app_server | peoplesoft_batch_server |
  peoplesoft_pia_server } <instance_name> [ --server_pool <serverpool> | --node
```

stop

The *agctl stop* command stops the Application Server, Batch Server, or PIA Server instance on Oracle Clusterware. Only the Oracle Clusterware applications that are starting or running are stopped.

```
agctl stop { peoplesoft_app_server | peoplesoft_batch_server |
             peoplesoft_pia_server } <instance_name> [ --force ]
```

status

Unlike the *agctl check* command, the *agctl status* command displays the current state of the Application Server, Batch Server, or PIA Server instance managed by Oracle Clusterware

```
agctl status { peoplesoft_app_server | peoplesoft_batch_server |
  peoplesoft_pia_server } <instance_name> [ --node <node_name> ]
```

PeopleSoft Server Connectivity

Application Server

If during *agctl add* or *modify* operations --*vip_name* option is added or modified or --*network, --ip, --user* options are added or modified *agctl* will display the following message:

> Use 'psadmin' command to configure the JOLT Listener Address to IP address <*ip-address*>

The PeopleSoft user must manually change the Application Server Configuration via the *psadmin* command.

Notes: The Application Server needs to be restarted for the changes to take effect.

Web PIA Server

If during *agctl add* or *modify* operations --*vip_name* option is added or modified or --*network, --ip, --user* options are added or modified *agctl* will display the following message:

> To maintain the web server configuration settings, go online to:
> PeopleTools > Web Profile > Web Profile Configuration
>
> Add IP address <*ip-address*> to the 'psserver' entry in the configuration.property file

The PeopleSoft user must manually change the Web Profile Configuration as documented in the PeopleSoft Administrator guide.

Notes: The PIA Server needs to be restarted for the changes to take effect.

Batch Server (Process Scheduler)

If during *agctl add* or *modify* operations --*vip_name* option is added or modified or --*network, --ip, --user* options are added or modified *agctl* will display the following message:

> Use *psadmin* command to configure the JOLT Listener Address to IP address <*ip-address*>

The PeopleSoft user must manually change the Batch Server Configuration via the *psadmin* command.

Notes: The Batch Server needs to be restarted for the changes to take effect.

MySQL

MySQL is the world's most popular open source database, enabling the cost-effective delivery of reliable, high-performance and scalable Web-based and embedded database applications. A MySQL Grid instance consists of the server software, **mysqld**, its configuration, **my.cnf,** its data dictionary, database(s) and logs; accessible through a virtual IP address. A MySQL Grid instance is started on an available node within the Grid according to resource policy settings. There is exactly one running copy of the MySQL instance per cluster, specifically cluster resource pool.

MySQL Enterprise Edition

MySQL Enterprise Edition includes the most comprehensive set of advanced features, management tools and technical support to achieve the highest levels of MySQL scalability, security, reliability, and uptime. It reduces the risk, cost, and complexity in developing, deploying, and managing business-critical MySQL applications. A MySQL Enterprise Edition subscription or license is required for use of the MySQL Agent for Oracle Clusterware.

Version Support Matrix

The following combinations of Clusterware and MySQL Enterprise Edition releases are supported.

Grid Infrastructure (GI)	MySQL
11.2.0.4+/12.1+/12.2+/18+/19+	5.6+

GI 11.2.0.4.+ release implies the same Oracle Automatic Storage Manager (ASM) release

Agent Functions

- Start the MySQL instance (server process)
- Monitor the MySQL instance (server process) and its dependent resources
- Initiate MySQL instance failover including relocation
- Stop the MySQL server process and
- Clean the MySQL instance of temporary files allowing restart

Resource Dependency Options

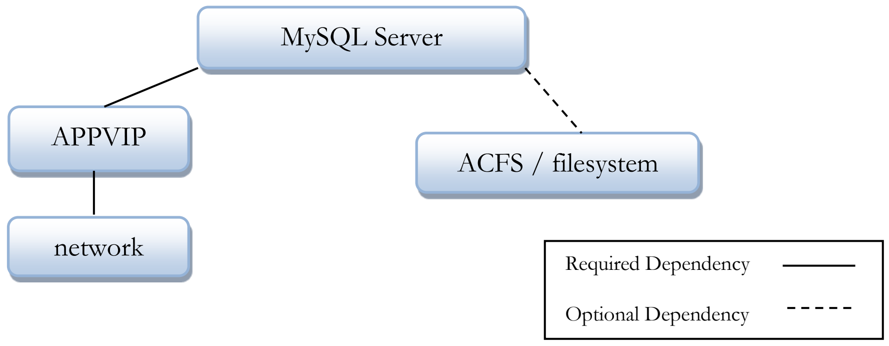


Figure 8

As indicated in Figure 8 above, the MySQL instance resource requires an application VIP (shown as APPVIP) for the manager connectivity by remote client applications and for both management and monitoring of the MySQL server components. An APPVIP requires a network resource, e.g., net.1.network, where 1 is the network id configured for this Grid deployment. MySQL also requires the existence of a shared disk, mounted on all nodes of the designated resource pool (nodes of the managed Grid on which the MySQL instance can be started). This most likely is provided by ACFS/ASM, but can be externally provided by any SAN or NFS shared storage arrangement.

Other MySQL Dependencies

In addition to the MySQL Server data directory, additional elements should be shared or synchronized between the nodes supporting the MySQL Grid deployment. This should include server configuration files, which can be deployed on shared storage and sym-linked from the appropriate standard location. Other possible shared dependencies include certain log files (e.g., MySQL Enterprise Audit Log output) or SSL certificates and keys.

State Definitions

ONLINE – The MySQL instance identified by its VIP is in a started mode and available for connection by any of its client interfaces, e.g., '*mysql*' command interface, Python connector, MemcacheD, '*mysqladmin*' administration command interface. Normal connection based query (SQL) can be initiated by a MySQL client. Note that an ONLINE status represents availability of the MySQL Server to new client connections. Availability of specific resources may be impacted by normally occurring timeouts will occur, as with table locks, binlog management locks,

OFFLINE – The MySQL instance identified by its VIP is not available for any connections.

INTERMEDIATE – The MySQL server process, *mysqld,* is running, but currently not responsive to connection requests. This is most commonly observed during startup and shutdown phases of MySQL Server, including any crash recovery.

UNKNOWN – This state indicates that Oracle Clusterware is unable to manage the named resource instance and manual Oracle Clusterware or MySQL intervention is required to correct the root cause. Once corrected, agctl start/stop commands should be used exclusively for managing the resource.

Resource Type Definitions

Oracle Clusterware uses the resource type, xag.mysql.type, to capture a set of attribute settings defining the MySQL instance. This type definition is installed at the time the bundled agents are installed.

Monitoring of MySQL server

Oracle Clusterware performs a basic monitoring of the MySQL resource instance using the *mysqladmin* client application. Complete monitoring of MySQL and associated processes still occurs through *mysqladmin,* the administration command interface, *MySQL Enterprise Monitor,* or other independent administration and performance monitors using any of MySQL Connector technologies to communicate with the MySQL server. Oracle Clusterware performs the external tasks of assuring defined dependencies are available on the assigned server, relocating a MySQL Server instance to an available node when dependent resources become unavailable and in general provide load balancing of managed applications across the Grid definition in a coordinated manner.

MySQL continues to record and report is operational progress, but the constant status polling of the Clusterware control process supports the general application stack integration, where dependent resources changing state result in commanded change or relocation of the higher level application.

Managing MySQL Service with systemd

As of MySQL 5.7.6, if you install MySQL using an RPM distribution on Linux platform, the server startup and shutdown are managed by *systemd.*

Oracle Clusterware performs basic monitoring of the MySQL resource instance using the *systemctl* command. Complete monitoring of MySQL and associated processes still occurs through the *systemctl* command. Oracle Clusterware performs the external tasks of assuring defined dependencies are available on the assigned server, relocating a MySQL Server instance to an available node when dependent resources become unavailable and in general provide load balancing of managed applications across the Grid definition in a coordinated manner.

Requirements for using systemctl Command to manage MySQL Server with systemd

The systemd system requires that services are managed using systemctl by the super-user (root). The user that executes the systemctl commands needs to have super-user privileges, unless the user has sudo access. The sudo command allows a system administrator to give a certain users the ability to run some or all commands as root. In order to register the MySQL Server systemd service with Oracle Clusterware, the sudo access is required for the mysql user.

The simplest change to get sudo to work is to add entry similar to the one below to the sudoers file (requires root privileges) e.g:
mysqluser ALL=NOPASSWD: /bin/systemctl start mysqld, /bin/systemctl stop mysqld, /bin/systemctl status mysqld

Clusterware Monitoring Credentials

Like other client applications, mysqladmin requires user credentials to establish a connection to MySQL Server. The Clusterware agent requires only the very limited USAGE privilege to execute necessary availability checks. A dedicated, limited-privilege account should be created for use by the Clusterware Agent, with connections accepted only from localhost. This account can be created using the following SQL command (where <PASSWORD> is replaced with the actual password):

```
mysql> CREATE USER oracle@localhost IDENTIFIED BY '<PASSWORD>';
```

The user credentials are not managed within Oracle Clusterware – users will need to define the credentials used by *mysqladmin* by adding a [mysqladmin] section to the configuration file, as shown below:

```
[mysqladmin]
user=oracle
password=<PASSWORD>
```

By storing the credentials in the configuration file, any system user with read privileges on the configuration file has access to the credentials.

Securing Monitoring Credentials Using Socket Peer-Credential Authentication

Access to the MySQL Server using account credentials for the Clusterware agent and can be further restricted to specific system accounts using the Socket Peer-Credential Authentication Plug-in for MySQL. By defining the MySQL user account using the same name as the system account under which the Clusterware agent runs, access can be restricted using the Socket Peer-Credential Authentication Plug-in to just that system user. Once the SocketPeer-Credential Authentication Plug-in is successfully installed, the MySQL Server user account can be created as follows:

```
mysql> CREATE USER oracle@localhost IDENTIFIED WITH auth_socket;
```

Assuming the Clusterware agent runs using the "oracle" OS user account, defining just the username (no password) in the MySQL configuration file will be sufficient to successfully connect using mysqladmin.

MySQL Enterprise Monitor Agent Deployment and Configuration

MySQL Enterprise Monitor (MEM) can monitor the MySQL Server instance remotely, as described in the MEM documentation. Because the HA aspect of the deployment is abstracted by the VIP, no special configuration is required here. However, this MEM deployment option prohibits collection of OS-level statistics for MEM consumption, as there is no MEM Agent running locally on the machine hosting MySQL Server instances.

MySQL Agent AGCTL Syntax

The *agctl add* command adds a MySQL Server instance to Oracle Clusterware. The MySQL admin can add the MySQL Server components and all of its options when adding the resources with pre-created VIP. The options *network_number, ip_address*, must be executed as root in order to create the VIP and the *user* and *group* options must be correctly defined.

The following are common *agctl* commands for the MySQL Server agent.

```
agctl [ add | modify ] mysql_server <instance_name>
      --mysql_home <MySQL software installation directory>
      --datadir <location of MySQL runtime data>
      --mysql_type MYSQL | MONITOR
      --mysql_lib <MySQL library directory>
      --monitor_agent <MySQL monitor Enterprise agent>
      --vip_name <network virtual ip name> | --network=<network_number>
                                             --ip=<virtual ip for resource>
                                             --user=<user>
                                             --group=<group>
      --serverpool <serverpool_name> | --nodes <node1>=[,…]
      --environment_vars <name>=<value>[,…]
```

MySQL Agent AGCTL Syntax for systemd-managed environments

The *agctl add* command adds MySQL Server instances to Oracle Clusterware. The MySQL admin can add the MySQL Server components and all of its options when adding the resources with pre-created VIP. The options *network_number, ip_address*, must be executed as root in order to create the VIP and the *user* and *group* options must be correctly defined.

The following are common *agctl* commands for the MySQL service systemd agent.

```
agctl [ add | modify ] mysql_server <instance_name>
      --service_name <service_name>
      --vip_name <network virtual ip name> | --network=<network_number>
                                             --ip=<virtual ip for resource>
                                             --group=<group>
      --serverpool <serverpool_name> | --nodes <node1>=[,…]
      --environment_vars <name>=<value>[,…]
```

Where the options for *agctl add* and *modify* commands for MySQL Server components are:

instance_name	A unique name labeling this MySQL Grid Instance. Required, unchangeable
mysql_home	The MySQL software installation directory. Required
datadir	The mounted shared disk where all MySQL dynamic data: database, transaction logs, and, configuration settings reside. Required
mysql_type	Declaration of managed resource type MYSQL server default. Additional values may be introduced in future releases.
mysql_lib	The folder where MySQL software libraries are installed. Defaults based on software install type

monitor_agent	The name of MySQL Monitor Enterprise Agent
service_name	MySQL service name. If it's not specified when registering the MySQL service, the default service name will be used. - For SLES system, the default is *mysql* - For Linux platforms, the default is *mysqld*
ip	The IP address by with network access to this MySQL resource instance is possible Either --ip or –vip_name is required.
vip_name	A logical name uniquely associated with the virtual IP, used if the IP resource has already been registered with *appvipcfg*
network	The network number if a new VIP resource is to be created Required
ip	The IP address by with network access to this MySQL Server resource instance is possible Either --ip or –vip_name is required.
user	The name of the MySQL OS user who owns the application VIP. Required if not using pre-configured VIP *The user who is adding the VIP resource and the owner of VIP resource are not necessary the same. For example, 'root' user can add the VIP resource, but its owner is foo user.*
group	The name of the MySQL OS group to which the MySQL user belongs.
serverpool	The name of the serverpool in which this MySQL instance should be started with all its required resources. If omitted, defaults to all nodes in the Grid
Nodes	A list of all nodes where this MySQL instance can be run. If omitted, defaults to all nodes in the Grid
environment_vars	An optional list of environment variables to be passed when the MySQL Server instance is started/stopped/monitored.

Note: The application IP can be created in advance by either the Grid Admin user using the *appvipcfg* command. Further, this command must run as *root* if the virtual IP is being created, not referenced, in this *agctl [add | modify]* command.

config

Use the *agctl config* command to list a MySQL resource instance configuration. It will also print a brief descriptive message listing resource attributes.

```
# agctl config mysql_server <instance_name>
```

enable

Use the *agctl enable* command to re-enable a MySQL resource. When added, a resource is automatically enabled.

```
# agctl enable mysql_server <instance_name>
```

disable

Use the *agctl disable* command to disable a MySQL resource. This will prevent the Clusterware from starting the instance.

```
# agctl disable mysql_server <instance_name>
```

relocate

Use the *agctl relocate* command to relocate a running MySQL resource from one server to another:

```
# agctl relocate mysql_server <instance_name> [--serverpool <serverpool_name> | --node <node_name>
```

remove

Use the *agctl remove* command to unregister a MySQL resource instance configuration from Oracle Clusterware:

```
# agctl remove mysql_server <instance_name> [--force]
```

start

Use the *agctl start* command to start a MySQL resource. The optional parameters serverpool and node are used to indicate desired placement at startup. An error is reported if the instance is not in an OFFLINE state:

```
# agctl start mysql_server <instance_name> [--serverpool <serverpool_name> | --node <node_name>]
```

stop

Use the *agctl stop* AGCTL command to stop a MySQL resource. An error is reported if the instance is not in an ONLINE state.

```
# agctl stop mysql_server <instance_name>
```

status

Use the *agctl status* command to return the current status the MySQL resource.

```
# agctl status mysql_server <instance_name>
```

Sample Configuration

The following is an example of a MySQL configuration on a 3-node cluster. The MySQL instance is registered with Clusterware using agctl, as root:

```
# agctl add mysql_server 192_168_56_111 –mysql_home /usr –datadir /u01/app/mysql/192_168_56_111 –
mysql_type MYSQL --network 1 –ip 192.168.56.111—user mysql –group dba-
```

Where

- The MySQL instance is “192_168_56_111”
- The MySQL software was installed in /usr, i.e., was installed using rpm –I MySQL-<package id>
- The MySQL instance type is MYSQL, indicating a server instance
- The nodes in the managed Grid cluster this instance can be started on are host1, host2 and host3.
- The network id 1 is the default, indicating *ora.net1.network*
- The virtual IP (either existing or created before the resource is added) is 192.168.56.111
- The user and group identifier the process(es) are run are ‘mysql’ and ‘dba’ (note agctl add allows parameters user and group for all instance types)

Sample Configuration for MySQL service systemd

The following is an example of a MySQL configuration on a 2-node cluster. The MySQL service systemd instance is registered with Clusterware using agctl, as root:

```
# agctl add mysql_server server101  –network 1 –ip 192.168.76.111 –user foo –group dba
```

Where

- The MySQL instance is “server101”
- The MySQL user is “foo” who owns the VIP and MySQL credentials
- The network id 1 is the default, indicating *ora.net1.network*
- The virtual IP (either existing or created before the resource is added) is 192.168.76.111
- The user and group identifier the process(es) are run are ‘foo’ and ‘dba’ (note agctl add allows parameters user and group for all instance types)

JD Edwards EnterpriseOne

JD Edwards was the very first ERP solution company and they got their name by building accounting ERP software for IBM minicomputers in the late 1970s. Today, JD Edwards offers a holistic enterprise solution, helping organizations manage every asset of their business, including financials, sales, inventory, manufacturing, human resources, customer relations and much more.

JD Edwards Enterprise Server

The JD Edwards EnterpriseOne Enterprise Server is a critical component of the EnterpriseOne architecture. It validates users (Security Server), runs Business Logic (Logic Server) and Batch processing (UBE's), Interoperability Logic (XML requests), Embedded BI, and many other critical functions for EnterpriseOne.

Version Support Matrix

The Oracle Grid Infrastructure for JD Edwards EnterpriseOne supports the following combination of Oracle Grid Infrastructure / JD Edwards EnterpriseOne releases. The respective Database versions must be compatible with the Grid Infrastructure and JD Edwards EnterpriseOne versions.

Grid Infrastructure	JD Edwards EnterpriseOne
11.2.0.4.+/12.1+/12.2+/18+/19+	9.1+

Agent Functions

- Manage the JD Edwards Enterprise Server failover, including the Batch and Logic Servers
- Start the JD Edwards Enterprise Server instance and its dependent resources
- Monitor the JD Edwards Enterprise Server and its dependent resources
- Initiate JD Edwards Enterprise Server instance failover including relocation
- Stop the JD Edwards Enterprise Server process and its dependent resources
- Clean the JD Edwards Enterprise Server instance of temporary files allowing restart

Resource Dependency Options

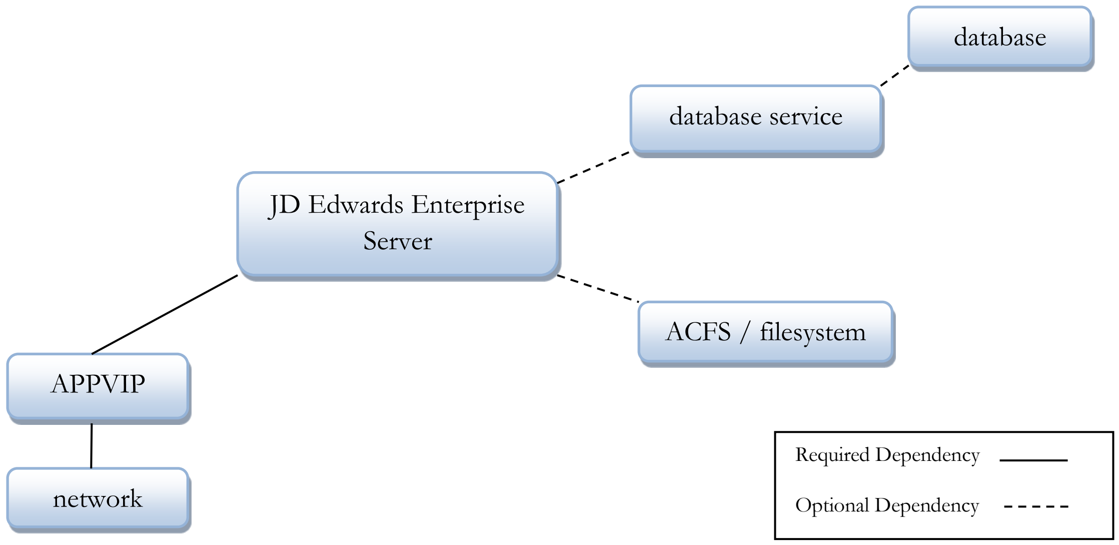


Figure 9

As indicated in Figure 9 above, the JD Edwards Enterprise Server instance resource requires a unique application VIP (shown as APPVIP) for the manager connectivity by remote client applications and for both management and monitoring of the Enterprise Server components. An APPVIP requires a network resource, e.g., net.1.network, where 1 is the network id configured for this Grid deployment. Other optional resource dependencies include ACFS or other supported file systems, and database connectivity dependencies either on the database service or the database directly.

JD Edwards Enterprise Server, Database, Database Service, and File System Dependencies

The following constraints will be used to establish the dependencies between Enterprise Server, Database, Database Service, and File system.

If the *databases, db_service,* and/or *filesystems* option is specified, the Enterprise Server will have a *hard(global:database,db_service,filesystem)* start dependency on Database, Database Service, and File System resource. A *hard* dependency means Database, Database Service, and File System resource must be running before Application Server can start. The *global* modifier means Application Server can start as long as the Database, Database Service, and file system resource are running on any node of the cluster.

State Definitions

ONLINE – The JD Edwards Enterprise Server instance identified by its VIP is in a started mode and available for connection by any of its client interfaces, e.g., '*RunOneWorld.sh*' command interface. Note that an ONLINE status represents availability of the Enterprise Server to new client connections.

OFFLINE – The Enterprise Server instance identified by its VIP is not available for any connections.

INTERMEDIATE – The Enterprise Server process is running, but currently not responsive to connection requests. This is most commonly observed during startup and shutdown phases of Enterprise Server, including any crash recovery.

UNKNOWN – This state indicates that Oracle Clusterware is unable to manage the named resource instance and manual Oracle Clusterware or Enterprise Server intervention is required to correct the root cause. Once corrected, agctl start/stop commands should be used exclusively for managing the resource.

Resource Type Definitions

Oracle Clusterware uses the resource type, xag.jde.type, to capture a set of attribute settings defining the Enterprise Server instance. This type definition is installed at the time the bundled agents are installed.

JD Edwards Agent AGCTL Syntax

The *agctl add* command adds JD Edwards Enterprise Server instances to Oracle Clusterware. The JD Edwards admin can add the JD Edwards Enterprise Server components and all of its options when adding the resources with pre-created VIP. The options *network_number, ip_address*, must be executed as root in order to create the VIP and the user and group options must be correctly defined.

The following are common *agctl* commands for the JD Edwards agent.

```
agctl [ add | modify ] jde_enterprise_server <instance_name>
       --jde_home <jde_home>
       --serverpool <serverpool_name> | --nodes <node1,node2, ...>
       --vip_name <network virtual ip name> | --network=<network_number> --ip=<virtual ip for resource>
                                         --user=<user> --group=<group>
       --attribute "<attribute_name=value,attribute_name2=value,...>"
       --filesystems <filesystem1[,...]>
       --databases <database1[,...]>
       --db_services <db_service1[,...]>
       --environment_vars "<name1=value1[,...]>
```

Where the options for *agctl add* and *modify* commands for JD Edwards components are:

instance_name	Name of the JD Edwards Enterprise Server instance. The *instance_name* must be unique and cannot be changed after the resource is registered. Required, unchangeable
jde_home	The path of the JD Edwards installed directory. Required
serverpool	The name of the server pool in which this Enterprise Server instance should be started with all its required resources. If omitted, defaults to all nodes in the Grid
nodes	A list of all nodes where this Enterprise Server instance can be run. If omitted, defaults to all nodes in the Grid.

vip_name	A logical name uniquely associated with the virtual IP, used if the IP resource has already been registered with *appvipcfg*
network	The network number if a new VIP resource is to be created Required
ip	The IP address by with network access to this Enterprise Server resource instance is possible Either --ip or –vip_name is required.
user	The name of the JD Edwards OS user who owns the application VIP. Required if not using pre-configured VIP *The user who is adding the VIP resource and the owner of VIP resource are not necessary the same. For example, 'root' user can add the VIP resource, but its owner is foo user.*
group	The name of the JD Edwards OS group to which the JD Edwards user belongs.
attribute	Sets/overrides default values for any attribute that can be applied to the Enterprise Server resource. For example: *--attribute "CHECK_INTERVAL=60,FAILURE_INTERVAL=30"* *Note: the time is measure in seconds*
filesystems	List of file system resource(s) that need to be mounted before the Enterprise Server can start. These would usually be the file system(s) containing the application home/binaries. *Note: The filesystems and db_services are mutually exclusive.*
databases	List of database instances created for Enterprise Server instance.
db_services	List of data base services(s) that need to be up and running before Enterprise Server can start. Note: The *filesystems* and *db_services* are mutually exclusive.
environment_vars	An optional list of environment variables to be passed when the JD Edwards Enterprise Server instance is started/stopped/monitored.

Note: The application IP can be created in advance by either the Grid Admin user using the *appvipcfg* command. Further, this command must run as *root* if the virtual IP is being created, not referenced, in this *agctl [add | modify]* command.

config

Use the *agctl config* command to list a Enterprise Server resource instance configuration. It will also print a brief descriptive message listing resource attributes.

Note: The *instance_name* is optional. If the *instance_name* is not specified, all configured instances will be displayed.

```
agctl config jde_enterprise_server [ <instance_name> ]
```

automatic startup, failover, or restart. The Oracle Clusterware application supporting Enterprise Server instance may be up or down to use this function. The default value is *enabled*.

If the Enterprise Server instance is already enabled, then the command is ignored. Enabled objects can be started, and disabled objects cannot be started:

```
agctl enable jde_enterprise_server <instance_name> [ --node <node_name> ]
```

disable

The *agctl disable* command disables the Enterprise Server instance. Use the *agctl disable* command when the Enterprise Server instance must shut down for maintenance. This will prevent the Clusterware from starting the instance. The disabled object does not automatically restart.

```
agctl disable jde_enterprise_server <instance_name> [ --node <node_name> ]
```

relocate

Use the *agctl relocate* command to relocate a running Enterprise Server resource from one server to another:

```
agctl relocate jde_enterprise_server <instance_name>
               [ --server_pool <serverpool> | --node <node_name> ]
```

remove

The *agctl remove* command removes the Enterprise Server configuration information from Oracle Clusterware. Environment settings for the object are also removed.

Notes: This command does not destroy the JD Edwards EnterpriseOne installed root directory.

Removing the Enterprise Server is allowed only if it has no dependency, unless the *--force* option is used.

```
agctl remove jde_enterprise_server <instance_name> [ --force ]
```

start

Use the *agctl start* command to start a Enterprise Server resource. The optional parameters serverpool and node are used to indicate desired placement at startup.

```
agctl start jde_enterprise_server <instance_name>
            [ --server_pool <serverpool> | --node <node_name> ]
```

stop

Use the *agctl stop* command to stop an Enterprise Server resource.

```
agctl stop jde_enterprise_server <instance_name> [ --force ]
```

status

Use the *agctl status* command to return the current status the Enterprise Server resource:

```
agctl status jde_enterprise_server <instance_name> [ --node <node_name> ]
```

Sample Configuration

The following is an example of a JD Edwards Enterprise Server configuration on a 2-node cluster. The Enterprise Server instance is registered with Clusterware using agctl, as root:

```
agctl add jde_enterprise_server jde_ent101 --jde_home /u01/jdedwards/e910
  --network 1 –ip 192.168.31.101 --user jde --group dba
```

Where

- The Enterprise Server instance is "jde_ent101"
- The JD Edwards EnterpriseOne software was installed in /u01/jdedwards/e910
- The nodes in the managed Grid cluster this instance can be started on are host1 and host2.
- The network id 1 is the default, indicating *ora.net1.network*
- The virtual IP (either existing or created before the resource is added) is 192.168.31.101
- The user and group identifier the process(es) are run are 'jde' and 'dba' (note: agctl add allows parameters user and group for all instance types)

WebLogic Administration Server

WebLogic is a server software application that runs on a middle tier, between back-end databases and related applications and browser-based thin clients. The WebLogic Administration Server serves as a central point of contact for server instances and system administration tools.

Version Support Matrix

The Oracle Grid Infrastructure for WebLogic Server supports the following combination of Oracle Grid Infrastructure / WebLogic Server releases. The respective Database versions must be compatible with the Grid Infrastructure and WebLogic Server versions.

Grid Infrastructure	WebLogic Server
11.2.0.4.+/12.1+/12.2+/18+/19+	11.0+

Agent Functions

- Manage the WebLogic Administration Server failover
- Start the WebLogic Administration Server instance and its dependent resources
- Monitor the WebLogic Administration Server and its dependent resources
- Initiate WebLogic Administration Server instance failover including relocation
- Stop the WebLogic Administration Server process and its dependent resources
- Clean the WebLogic Administration Server instance of temporary files allowing restart

Resource Dependency Options

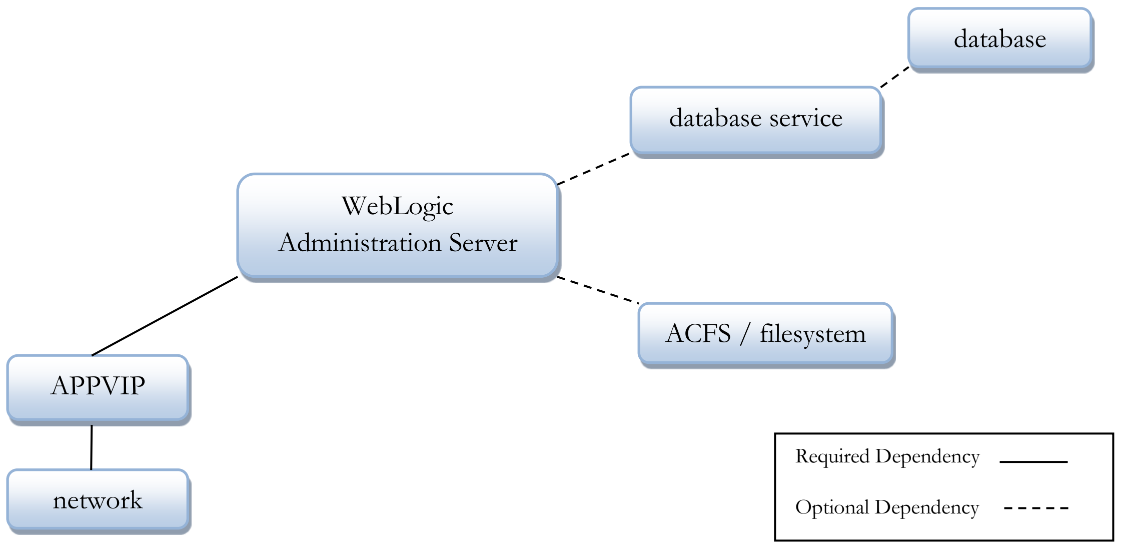


Figure 10

As indicated in Figure 10 above, the WebLogic Administration Server instance resource requires a unique application VIP (shown as APPVIP) for the manager connectivity by remote client applications and for both management and monitoring of the Administration Server components. An APPVIP requires a network resource, e.g., net.1.network, where 1 is the network id configured for this Grid deployment. Other optional resource dependencies include ACFS or other supported file systems, and database connectivity dependencies either on the database service or the database directly.

WebLogic Administration Server, Database, Database Service, and File System Dependencies

The following constraints will be used to establish the dependencies between WebLogic Administration Server, Database, Database Service, and File system.

If the *databases, db_service,* and/or *filesystems* option is specified, the Administration Server will have a *hard(global:database,db_service,filesystem)* start dependency on Database, Database Service, and File System resource. A *hard* dependency means Database, Database Service, and File System resource must be running before Application Server can start. The *global* modifier means Application Server can start as long as the Database, Database Service, and file system resource are running on any node of the cluster.

State Definitions

ONLINE – The Administration Server instance identified by its VIP is in a started mode and available for connection by any of its client interfaces, e.g., '*startWeblogic.sh*' command interface. Note that an ONLINE status represents availability of the Administration Server to new client connections.

OFFLINE – The Administration Server instance identified by its VIP is not available for any connections.

INTERMEDIATE – The Administration Server process is running, but currently not responsive to connection requests. This is most commonly observed during startup and shutdown phases of Administration Server, including any crash recovery.

UNKNOWN – This state indicates that Oracle Clusterware is unable to manage the named resource instance and manual Oracle Clusterware or Administration Server intervention is required to correct the root cause. Once corrected, agctl start/stop commands should be used exclusively for managing the resource.

Resource Type Definitions

Oracle Clusterware uses the resource type, xag.wl.type, to capture a set of attribute settings defining the WebLogic Administration Server instance. This type definition is installed at the time the bundled agents are installed.

Monitoring of WebLogic Administration Server

Oracle Clusterware performs a basic monitoring of the WebLogic resource instance using the *startWeblogic.sh* client application. Complete monitoring of WebLogic and associated processes still occurs through *state(),* the WLST (WebLogic Scripting Tool) command interface. Oracle Clusterware performs the external tasks of assuring defined dependencies are available on the assigned server, relocating a WebLogic Server instance to an available node when dependent resources become unavailable and in general provide load balancing of managed applications across the Grid definition in a coordinated manner.

WebLogic continues to record and reports operational progress, but the constant status polling of the Clusterware control process supports the general application stack integration, where dependent resources changing state result in commanded change or relocation of the higher level application.

Clusterware Monitoring Credentials

The WLST *state()* command requires user credentials to check for the state of the Administration Server. The Clusterware agent requires only the very limited USAGE privilege to execute necessary availability checks. A dedicated, limited-privilege account should be created for use by the Clusterware Agent. This account can be created using the following WebLogic Admin command:

```
java weblogic.Admin -username <username> -userconfigfile <config-file>
-userkeyfile <key-file> STOREUSERCONFIG
```

Where

- The *username* is the WebLogic Administration Server username.
- The *config-file* is a file pathname at which the `STOREUSERCONFIG` command creates a user-configuration file. The pathname can be absolute or relative to the directory from which the command is entered.
- The *key-file* is a file pathname at which the `STOREUSERCONFIG` command creates a key file. The pathname can be absolute or relative to the directory from which the command is entered.

Enabling Auto Login by Using the Boot Identity File

A boot identity file contains user credentials for starting and stopping an instance of WebLogic Server. An Administration Server can refer to this file for user credentials instead of prompting for username and password. The boot identity file can be created using the following procedures:

1) Browse to *$DOMAIN_HOME/servers/<admin-server>/security* and edit the *boot.properties* file.

   Note: The *security* folder and the *boot.properties* file incase the parent folder is empty.

2) Modify the *boot.properties* file with the following values and save it.

   username=<*encrypted-username*>
   password=< *encrypted-password*>

3) The *encrypted-username* and *encrypted-password* can be created using the following WebLogic Java Server Utility:

```
java weblogic.security.Encrypt username

java weblogic.security.Encrypt password
```

Example:

```
$ java weblogic.security.Encrypt oracle
{AES}DQc8jgSpsBUGNyNfOqatRGr4PVaBFkplxlRFotFoycw=

$ java weblogic.security.Encrypt welcome1
{AES}Ls+9VzNYGbred9MT/ftNNiuUcgq7ss5nmSkxHFLu9Y8=

$ cat security/boot.properties
#Thu Jul 26 05:44:54 EDT 2012
password={AES}Ls+9VzNYGbred9MT/ftNNiuUcgq7ss5nmSkxHFLu9Y8=
username={AES}DQc8jgSpsBUGNyNfOqatRGr4PVaBFkplxlRFotFoycw=
```

WebLogic Agent AGCTL Syntax

The *agctl add* command adds WebLogic Administration Server instances to Oracle Clusterware. The WebLogic admin can add the WebLogic Administration Server components and all of its options when adding the resources with pre-created VIP. The options *network_number, ip_address*, must be executed as root in order to create the VIP and the user and group options must be correctly defined.

The following are common *agctl* commands for the WebLogic Server agent.

```
agctl [ add | modify ] weblogic_admin_server <instance_name>
       --domain_home <domain_home>
       --userconfigfile <userconfigfile>
       --userkeyfile <userkeyfile>
       --serverpool <serverpool_name> | --nodes <node1,node2, …>
       --vip_name <network virtual ip name> | --network=<network_number> --ip=<virtual ip for resource>
                                            --user=<user> --group=<group>
       --attribute "<attribute_name=value,attribute_name2=value, …>"
       --filesystems <filesystem1[,…]>
       --databases <database1[,…]>
       --db_services <db_service1[,…]>
       --environment_vars "<name1=value1[,…]>
```

Where the options for *agctl add* and *modify* commands for WebLogic Server components are:

instance_name	Name of the WebLogic Administration Server instance. The *instance_name* must be unique and cannot be changed after the resource is registered. Required, unchangeable
domain_home	The location of the WebLogic Server Domain. Required
userconfigfile	The file pathname at which the STOREUSERCONFIG command creates a user-configuration file. Required
Userkeyfile	The file pathname at which the STOREUSERCONFIG command creates a key file. Required
serverpool	The name of the server pool in which this Administration Server instance should be started with all its required resources. If omitted, defaults to all nodes in the Grid.
nodes	A list of all nodes where this Administration Server instance can be run. If omitted, defaults to all nodes in the Grid.
vip_name	A logical name uniquely associated with the virtual IP, used if the IP resource has already been registered with *appvipcfg*
network	The network number if a new VIP resource is to be created Required

ip	The IP address by with network access to this Administration Server resource instance is possible Either --ip or –vip_name is required.
user	The name of the WebLogic OS user who owns the application VIP. Required if not using pre-configured VIP *The user who is adding the VIP resource and the owner of VIP resource are not necessary the same. For example, 'root' user can add the VIP resource, but its owner is foo user.*
group	The name of the WebLogic OS group to which the WebLogic user belongs.
attribute	Sets/overrides default values for any attribute that can be applied to the Administration Server resource. For example: *--attribute "CHECK_INTERVAL=60,FAILURE_INTERVAL=30"* *Note: the time is measure in seconds*
filesystems	List of file system resource(s) that need to be mounted before the Administration Server can start. These would usually be the file system(s) containing the application home/binaries. *Note: The filesystems and db_services are mutually exclusive.*
databases	List of database instances created for Administration Server instance.
db_services	List of data base services(s) that need to be up and running before Administration Server can start. Note: The *filesystems* and *db_services* are mutually exclusive.
environment_vars	An optional list of environment variables to be passed when the WebLogic Administration Server instance is started/stopped/monitored.

Note: The application IP can be created in advance by either the Grid Admin user using the *appvipcfg* command. Further, this command must run as *root* if the virtual IP is being created, not referenced, in this *agctl [add | modify]* command.

config

Use the *agctl config* command to list a Administration Server resource instance configuration. It will also print a brief descriptive message listing resource attributes.

Note: The *instance_name* is optional. If the *instance_name* is not specified, all configured instances will be displayed.

```
agctl config weblogic_admin_server [ <instance_name> ]
```

enable

The *agctl enable* command enables the Administration Server instance so that it can run under Oracle Clusterware for automatic startup, failover, or restart. The Oracle Clusterware application supporting Administration Server instance may be up or down to use this function. The default value is *enabled*.

If the Administration Server instance is already enabled, then the command is ignored. Enabled objects can be started, and disabled objects cannot be started:

```
agctl enable weblogic_admin_server <instance_name> [ --node <node_name> ]
```

disable

The *agctl disable* command disables the Enterprise Server instance. Use the *agctl disable* command when the Administration Server instance must shut down for maintenance. This will prevent the Clusterware from starting the instance. The disabled object does not automatically restart.

```
agctl disable weblogic_admin_server <instance_name> [ --node <node_name> ]
```

relocate

Use the *agctl relocate* command to relocate a running Administration Server resource from one server to another:

```
agctl relocate weblogic_admin_server <instance_name>
               [ --server_pool <serverpool> | --node <node_name> ]
```

remove

The *agctl remove* command removes the Administration Server configuration information from Oracle Clusterware. Environment settings for the object are also removed.

Notes: This command does not destroy the WebLogic Server installed root directory.

Removing the Administration Server is allowed only if it has no dependency, unless the *--force* option is used.

```
agctl remove weblogic_admin_server <instance_name> [ --force ]
```

start

Use the *agctl start* command to start a Administration Server resource. The optional parameters serverpool and node are used to indicate desired placement at startup.

```
agctl start weblogic_admin_server <instance_name>
            [ --server_pool <serverpool> | --node <node_name> ]
```

stop

Use the *agctl stop* command to stop an Administration Server resource.

```
agctl stop weblogic_admin_server <instance_name> [ --force ]
```

status

Use the *agctl status* command to return the current status the Administration Server resource:

```
agctl status weblogic_admin_server <instance_name> [ --node <node_name> ]
```

Sample Configuration

The following is an example of a WebLogic Administration Server configuration on a 2-node cluster. The Administration Server instance is registered with Clusterware using agctl, as root:

```
agctl add weblogic_admin_server admin101
  --domain_home /u01/Oracle/Middleware/user_projects/domains/base_domain
  --userconfigfile /home/oracle/myconfig
  --userkeyfile /home/oracle/mykey
  --network 1 --ip 192.168.31.101 --user wl --group dba
```

Where

- The Administration Server instance is "admin101"
- The WebLogic Server software was installed in /u01/Oracle/Middleware
- The nodes in the managed Grid cluster this instance can be started on are host1 and host2.
- The network id 1 is the default, indicating *ora.net1.network*
- The virtual IP (either existing or created before the resource is added) is 192.168.31.101
- The user and group identifier the process(es) are run are 'wl' and 'dba' (note: agctl add allows parameters user and group for all instance types)

WebLogic Agent Notes

1. During *agctl add* or *modify* operations where the *--vip_name* option is added or modified or *--network, --ip,* or *–user* options are added or modified, the WebLogic user is responsible for updating the Listen Address of the Administration Configuration on every node to use the new VIP.
2. Control of the WebLogic Administration Server components must be through *agctl start/stop/status/relocate* commands. Do not stop the Administration Server processes via the WebLogic command line interface or via any other means as this will initiate the component failover.
3. Oracle Clusterware will start, stop, monitor, restart, and failover the Administration Server process. Clusterware will not start, stop or restart individual Managed Server processes.

Oracle E-Business Suite

The Oracle E-Business Suite Architecture is a framework for multi-tiered, distributed computing that supports Oracle E-Business Suite products. The E-Business Suite is a server software application that runs on the application tier, between back-end databases and browser-based thin clients.

Oracle E-Business Suite Concurrent Manager

The Concurrent Manager functions as the “boss" of all the other managers. It starts up, verifies the status, resets, and shuts down the individual managers. There is exactly one running Concurrent Manager per installation of E-Business Suite.

Version Support Matrix

The Oracle Grid Infrastructure for E-Business Suite supports the following combination of Oracle Grid Infrastructure / E-Business Suite releases. The respective Database versions must be compatible with the Grid Infrastructure and E-Business Suite versions.

Grid Infrastructure	E-Business Suite
12.1+/12.2+/18+/19+	12.0+

Agent Functions

- Manage the E-Business Suite Concurrent Manager failover
- Start the E-Business Suite Concurrent Manager instance and its dependent resources
- Monitor the E-Business Suite Concurrent Manager and its dependent resources
- Initiate E-Business Suite Concurrent Manager instance failover including relocation
- Stop the E-Business Suite Concurrent Manager process and its dependent resources
- Clean the E-Business Suite Concurrent Manager instance of temporary files allowing restart

Resource Dependency Options

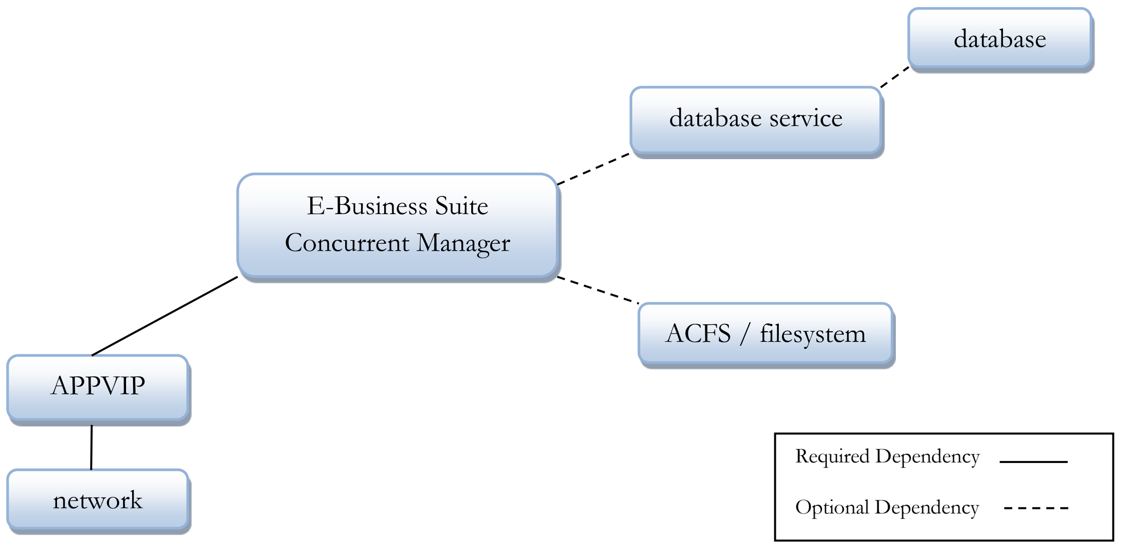


Figure 11

As indicated in Figure 11 above, the E-Business Suite Concurrent Manager instance resource requires a unique application VIP (shown as APPVIP) for the manager connectivity by remote client applications and for both management and monitoring of the Administration Server components. An APPVIP requires a network resource, e.g., net.1.network, where 1 is the network id configured for this Grid deployment. Other optional resource dependencies include ACFS or other supported file systems, and database connectivity dependencies either on the database service or the database directly.

E-Business Suite Concurrent Manager, Database, Database Service, and File System Dependencies

The following constraints will be used to establish the dependencies between E-Business Suite Concurrent Manager, Database, Database Service, and File system.

If the *databases, db_service,* and/or *filesystems* option is specified, the Concurrent Manager will have a *hard(global:database,db_service,filesystem)* start dependency on Database, Database Service, and File System resource. A *hard* dependency means Database, Database Service, and File System resource must be running before Application Server can start. The *global* modifier means Application Server can start as long as the Database, Database Service, and file system resource are running on any node of the cluster.

State Definitions

ONLINE – The Concurrent Manager instance identified by its VIP is in a started mode and available for connection by any of its client interfaces, e.g., '*adcmctl.sh'* command interface. Note that an ONLINE status represents availability of the Administration Server to new client connections.

OFFLINE – The Concurrent Manager instance identified by its VIP is not available for any connections.

INTERMEDIATE – The Concurrent Manager process is running, but currently not responsive to connection requests. This is most commonly observed during startup and shutdown phases of Concurrent Manager, including any crash recovery.

UNKNOWN – This state indicates that Oracle Clusterware is unable to manage the named resource instance and manual Oracle Clusterware or Concurrent Manager intervention is required to correct the root cause. Once corrected, agctl start/stop commands should be used exclusively for managing the resource.

Resource Type Definitions

Oracle Clusterware uses the resource type, xag.ebs.type, to capture a set of attribute settings defining the E-Business Suite Concurrent Manager instance. This type definition is installed at the time the bundled agents are installed.

Monitoring of E-Business Suite Concurrent Manager

Oracle Clusterware performs a basic monitoring of the E-Business Suite Concurrent Manager resource instance using the *adcmctl.sh* client application. Complete monitoring of E-Business Suite and associated processes still occurs through the *adcmctl.sh,* or the administration command interface. Oracle Clusterware performs the external tasks of assuring defined dependencies are available on the assigned server, relocating a Concurrent Manager instance to an available node when dependent resources become unavailable and in general provide load balancing of managed applications across the Grid definition in a coordinated manner.

The E-Business Suite continues to record and reports operational progress, but the constant status polling of the Clusterware control process supports the general application stack integration, where dependent resources changing state result in commanded change or relocation of the higher level application.

Enabling Auto Login by Using the Clusterware Credentials

Clusterware credentials management is the process by which the operating system receives the credentials from the user and secures that information for future use to the authenticating target. In the case of the E-Business Suite, the authenticating target is the E-Business Applications login.

The Oracle E-Business Suite requires the APPS password to start and stop the Concurrent Manager.
The Clusterware Credentials can be used to start and stop the Concurrent Manager without requiring the user to know the APPS password.

To use the Clusterware Credentials, the *--ebs_credentials* option must be used when registering the Concurrent Manager to the Oracle Clusterware. This allows the Clusterware Credentials to secure the E-Business Suite APPS login name and password. The *agctl add or modify ebs_concurrent_manager --ebs_credentials* commands must be run as **root**.

Example:

```
# agctl add ebs_concurrent_manager mycm
--instance_home /u01/install/VISION/fs1/inst/apps/EBSDB_ebs
--ebs_credentials --user foo
Enter the APPS username:
Enter the APPS password:

# agctl modify ebs_concurrent_manager mycm --ebs_credentials
Enter the APPS username:
Enter the APPS password:

Note: The --user option is unnecessary when modify EBS Credentials
```

Where

- The *–user* is the Operating System user that owns the instance or owns the E-Business Suite credentials
- The *username* is the E-Business APPS user
- The *password* is the E-Business APPS password

Note: *The user who is adding the E-Business Suite resource and the owner of E-Business Suite resource are not necessary the same. For example, 'root' user can add the E-Business Suite resource, but its owner is foo user.*

E-Business Suite Agent AGCTL Syntax

The *agctl add* command adds E-Business Suite Concurrent Manager instances to Oracle Clusterware. The E-Business admin can add the Concurrent Manager components and all of its options when adding the resources with pre-created VIP. The options *network_number, ip_address*, must be executed as root in order to create the VIP and the user and group options must be correctly defined.

The following are common *agctl* commands for the E-Business Suite Concurrent Manager Server agent.

```
agctl [ add | modify ] ebs_concurrent_manager <instance_name>
       --instance_home <inst_top>
       --ebs_credentials --user <user>
       --startup_params <name>=<value>[,…]
       --serverpool <serverpool_name> | --nodes <node1>=[,…]
       --vip_name <network virtual ip name> | --network=<network_number>
                                               --ip=<virtual ip for resource>
                                               --group=<group>
       --attribute <attribute_name>=<value>[,…]
       --filesystems <filesystem>[,…]
       --databases <database>[,…]
       --db_services <db_service>[,…]
       --environment_vars <name>=<value>[,…]
```

Where the options for *agctl add* and *modify* commands for E-Business Suite Concurrent Manager components are:

instance_name	Name of the E-Business Suite Concurrent Manager instance. The *instance_name* must be unique and cannot be changed after the resource is registered. Required, unchangeable
instance_home	The path of the E-Business Suite installation directory. Required
ebs_credentials	The Named Credentials for E-Business Suite Applications Login. Assuming the user already has access to the E-Business Suite database. This option requires super user privilege. Required
user	For EBS credentials, it's the name of the E-Business Suite OS user who owns the E-Business Suite credentials. Required Note: The –user option is unnecessary when modify the EBS credentials. For application VIP, it's the name of the E-Business Suite OS user who owns the application VIP. Required if not using pre-configured VIP *The user who is adding the VIP resource and the owner of VIP resource are not necessary the same. For example, 'root' user can add the VIP resource, but its owner is foo user.*
group	The name of the E-Business Suite OS group to which the E-Business Suite user belongs.

serverpool	The name of the server pool in which this Concurrent Manager instance should be started with all its required resources. If omitted, defaults to all nodes in the Grid.
nodes	A list of all nodes where this Concurrent Manager instance can be run. If omitted, defaults to all nodes in the Grid.
vip_name	A logical name uniquely associated with the virtual IP, used if the IP resource has already been registered with *appvipcfg*
network	The network number if a new VIP resource is to be created Required
ip	The IP address by with network access to this Concurrent Manager resource instance is possible Either --ip or –vip_name is required.
attribute	Sets/overrides default values for any attribute that can be applied to the Administration Server resource. For example: *--attribute "CHECK_INTERVAL=60,FAILURE_INTERVAL=30"* *Note: the time is measure in seconds*
filesystems	List of file system resource(s) that need to be mounted before the Concurrent Manager can start. These would usually be the file system(s) containing the application home/binaries. *Note: The filesystems and db_services are mutually exclusive.*
databases	List of database instances created for Concurrent Manager instance.
db_services	List of data base services(s) that need to be up and running before Concurrent Manager can start. Note: The *filesystems* and *db_services* are mutually exclusive.
environment_vars	An optional list of environment variables to be passed when the Concurrent Manager instance is started/stopped/monitored.

Note: The application IP can be created in advance by either the Grid Admin user using the *appvipcfg* command. Further, this command must run as *root* if the virtual IP is being created, not referenced, in this *agctl [add | modify]* command.

config

Use the *agctl config* command to list a Concurrent Manager resource instance configuration. It will also print a brief descriptive message listing resource attributes.

Note: The *instance_name* is optional. If the *instance_name* is not specified, all configured instances will be displayed.

```
agctl config ebs_concurrent_manager [ <instance_name> ]
```

enable

The *agctl enable* command enables the Concurrent Manager instance so that it can run under Oracle Clusterware for automatic startup, failover, or restart. The Oracle Clusterware application supporting Concurrent Manager instance may be up or down to use this function. The default value is *enabled*.

If the Concurrent Manager instance is already enabled, then the command is ignored. Enabled objects can be started, and disabled objects cannot be started:

```
agctl enable ebs_concurrent_manager <instance_name> [ --node <node_name> ]
```

disable

The *agctl disable* command disables the Concurrent Manager instance. Use the *agctl disable* command when the Concurrent Manager instance must shut down for maintenance. This will prevent the Clusterware from starting the instance. The disabled object does not automatically restart.

```
agctl disable ebs_concurrent_manager <instance_name> [ --node <node_name> ]
```

relocate

Use the *agctl relocate* command to relocate a running Concurrent Manager resource from one server to another:

```
agctl relocate ebs_concurrent_manager <instance_name>
               [ --server_pool <serverpool> | --node <node_name> ]
```

remove

The *agctl remove* command removes the Concurrent Manager configuration information from Oracle Clusterware. Environment settings for the object are also removed.

Notes: This command does not destroy the E-Business Suite installed root directory.

Removing the Concurrent Manager is allowed only if it has no dependency, unless the *--force* option is used.

```
agctl remove ebs_concurrent_manager <instance_name> [ --force ]
```

start

Use the *agctl start* command to start a Concurrent Manager resource. The optional parameters serverpool and node are used to indicate desired placement at startup.

```
agctl start ebs_concurrent_manager <instance_name>
            [ --server_pool <serverpool> | --node <node_name> ]
```

stop

Use the *agctl stop* command to stop a Concurrent Manager resource.

```
agctl stop ebs_concurrent_manager <instance_name> [ --force ]
```

status

Use the *agctl status* command to return the current status the Concurrent Manager resource:

```
agctl status ebs_concurrent_manager <instance_name> [ --node <node_name> ]
```

Sample Configuration

The following is an example of a E-Business Suite Concurrent Manager configuration on a 2-node cluster. The Administration Server instance is registered with Clusterware using agctl, as root:

```
agctl add ebs_concurrent_manager mycm
  --instance_home /u01/install/VISION/fs1/inst/apps/EBSDB_ebs
  --ebs_credentials --user foo
  --network 1 --ip 192.168.31.101
  --nodes host1,host2
```

Where

- The Concurrent Manager instance is "mycm"
- The E-Business Suite software was installed in /u01/install/VISION/fs1/inst/apps/EBSDB_ebs . The nodes in the managed Grid cluster this instance can be started on are host1 and host2.
- Create EBS credentials and owns by user *foo*
- The network id 1 is the default, indicating *ora.net1.network*
- The virtual IP (either existing or created before the resource is added) is 192.168.31.101

E-Business Suite Agent Notes

4. During *agctl add* or *modify* operations where the --*vip_name* option is added or modified or --*network*, --*ip*, or –*user* options are added or modified, the E-Business Suite user is responsible for updating the Listen Address of the Administration Configuration on every node to use the new VIP.

5. Control of the Concurrent Manager components must be through *agctl start/stop/status/relocate* commands. Do not stop the Concurrent Manager processes via the E-Business Suite command line interface or via any other means as this will initiate the component failover.
6. Oracle Clusterware will start, stop, monitor, restart, and failover the Concurrent Manager process. Clusterware will not start, stop or restart individual Managed Server processes.

Diagnostics and Troubleshooting

Diagnostic logs exist in each tier of the environment and should be evaluated to correctly identify the fault and root cause. The Oracle Clusterware may fail an application resource when the underlying application has been modified outside of AGCTL control. Under such circumstances, the agent may not have the modified application configuration and the start/stop or check actions would fail. The application configuration must be consistent with the AGCTL configuration of the resource. The Oracle Grid Infrastructure will report a fault attributed to another tier (application/network/OS) due to modifications that are inconsistent with the expected configuration.

Error Diagnostic Facility

The Oracle Grid Infrastructure Agents provide an `oerr xag <errno>` diagnostic facility for error analysis. To use this diagnostic facility, configure $XAGHOME/bin in the admin path. The Oracle RDBMS also uses the oerr facility for error diagnostics. The Oracle RDBMS `oerr` will only report on RDBMS errors. Please execute the oerr facility located in the $XAGHOME/bin directory for XAG related error messaging.

Log files

The xagsetup.sh will log local and remote installation execution to the log file: $XAG_HOME/log/<hostname>/xagsetup_<timestamp>.log. If xagsetup.sh terminates as a result of a failure in the initial stages of installation, this log file will be located in the /tmp directory

AGCTL and agent application execution is logged in separate files in the directory:

$XAG_HOME/log/<hostname>

The AGCTL log file will log all execution related to AGCTL commands for a given application. The xag_application.log will also log corresponding execution on a given Oracle Clusteware application resource.

Oracle Clusterware diagnostic logs are located in the $GRID_HOME/log/<hostname> directory. The most common logs to evaluate would be the alter<hostname>.log which is the overall status of the Clusterware and would include diagnostic data related to application agent activity. The other diagnostic data should be evaluated with guidance from Oracle Support.

Diagnostics Collection Script

When an Oracle Grid Infrastructure Agents error occurs, run the **xagdiagcollection.pl** diagnostics collection script to collect diagnostic information from Oracle Grid Infrastructure Agents into trace files. The diagnostics provide additional information so My Oracle Support can resolve problems. Run this script from the $XAG_HOME directory.

Syntax

Use the xagdiagcollection.pl script with the following syntax:

```
xagdiagcollection.pl {--collect [--xaghome path ]} [--clean]
```

Notes: The **xagdiagcollection.pl** script arguments are all preceded by two dashes (--).

Parameter	**Description**
--collect	Collects Oracle Grid Infrastructure Agents diagnostic information.
--xaghome *path*	Use this argument to override the location of the Oracle Grid Infrastructure Agents home (xag_home) **Note**: The **xagdiagcollection.pl** script derives the location of the Oracle Grid Infrastructure Agents home from the environment variable (XAG_HOME), so this argument is not required.
--clean	Use this parameter to clean up the diagnostic information gathered by the **xagdiagcollection.pl** script. **Note**: You cannot use this parameter with `-collect.`

Oracle Grid Infrastructure Cluster Health Monitor

The Oracle Grid Infrastructure ships with an integrated system health monitor that collects and reports on cluster wide system metrics that affect the performance and availability of the cluster. These metrics are captured and can be used for offline replay for performance diagnostics or root cause analysis. Please see the *Oracle Clusterware Administration and Deployment Guide 11g Release 2 (11.2) Appendix H Troubleshooting* for more details.

Resource profile

Application administrators have read/write access for application resources under their control. The output from `crsctl status resource <resource name> -t` would provide run time status. The complete profile for an application resource can be viewed with $GRID_HOME/bin/crsctl status resource<resource name> -f` which would produce output similar to the attribute key/value pairs listed above for each application resource.

Support

The Oracle Grid Infrastructure Agents are fully supported by the Oracle Support organization. Supplementary documentation is available on My Oracle Support.

Oracle Grid Infrastructure Agents
(Standalone and Bundled)
October, 2019
Authors: John P. McHugh, Jonathan Pham, Shankar Iyer , Andrey Gusev, Ian Cookson, Sourav Bhattacharya, Burt Clouse, Joe Debuzna, Chris Osterdoc, Tracy West, Todd Farmer

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200

oracle.com