CHAPTER 3

Administering GlassFish Server Clusters

A *cluster* is a collection of GlassFish Server instances that work together as one logical entity. A cluster provides a runtime environment for one or more Java Platform, Enterprise Edition (Java EE) applications. A cluster provides high availability through failure protection, scalability, and load balancing.

The Group Management Service (GMS) enables instances to participate in a cluster by detecting changes in cluster membership and notifying instances of the changes. To ensure that GMS can detect changes in cluster membership, a cluster's GMS settings must be configured correctly.

The following topics are addressed here:

- “About GlassFish Server Clusters” on page 31
- “Group Management Service” on page 32
- “Creating, Listing, and Deleting Clusters” on page 40

About GlassFish Server Clusters

A *cluster* is a named collection of GlassFish Server instances that share the same applications, resources, and configuration information. For information about GlassFish Server instances, see Chapter 4, “Administering GlassFish Server Instances.”

GlassFish Server enables you to administer all the instances in a cluster as a single unit from a single host, regardless of whether the instances reside on the same host or different hosts. You can perform the same operations on a cluster that you can perform on an unclustered instance, for example, deploying applications and creating resources.

A cluster provides high availability through failure protection, scalability, and load balancing.

- **Failure protection.** If an instance or a host in a cluster fails, GlassFish Server detects the failure, redirects requests from the failed instance to other instances in the cluster, and recovers the user session state. Because the same applications and resources are on all instances in the cluster, an instance can fail over to any other instance in the cluster.

  To enable the user session state to be recovered, each instance in a cluster sends in-memory state data to another instance. As state data is updated in any instance, the data is replicated.
- **Scalability.** If increased capacity is required, you can add instances to a cluster with no disruption in service. When an instance is added or removed, the changes are handled automatically.
- **Load balancing.** If instances in a cluster are distributed among different hosts, the workload can be distributed among the hosts to increase overall system throughput.

Group Management Service

The Group Management Service (GMS) is an infrastructure component that is enabled for the instances in a cluster. When GMS is enabled, if a clustered instance fails, the cluster and the Domain Administration Server (DAS) are aware of the failure and can take action when failure occurs. Many features of GlassFish Server depend upon GMS. For example, GMS is used by the in-memory session replication, transaction service, and timer service features.

If server instances in a cluster are located on different machines, ensure that all the server instance machines and the DAS machine are on the same subnet and that multicast is enabled for the network. To test whether multicast is enabled, use the `validate-multicast`(1) subcommand.

GMS is a core service of the Shoal framework. For more information about Shoal, visit the Project Shoal home page (`https://shoal.dev.java.net/`).

The following topics are addressed here:

- "GMS Configuration Settings" on page 32
- "Dotted Names for GMS Settings" on page 34
- "To Preconfigure Nondefault GMS Configuration Settings" on page 35
- "To Change GMS Settings After Cluster Creation" on page 36
- "To Check the Health of Instances in a Cluster" on page 37
- "To Validate that Multicast Transport Is Available for a Cluster" on page 37
- "Using the Multi-Homing Feature With GMS" on page 38

GMS Configuration Settings

GlassFish Server has two types of GMS settings:

- *GMS cluster settings* — These are determined during cluster creation. For more information about these settings, see "To Create a Cluster" on page 40.

- *GMS configuration settings* — These are determined during configuration creation and are explained here.

The following *GMS configuration settings* are used in GMS for group discovery and failure detection:

`group-discovery-timeout-in-millis`
: Indicates the amount of time an instance's GMS module will wait during instance startup for discovering other members of the group.

  The `group-discovery-timeout-in-millis` timeout value should be set to the default or higher. The default is 5000.

`max-missed-heartbeats`
: Indicates the maximum number of missed heartbeats that the health monitor counts before the instance can be marked as a suspected failure. GMS also tries to make a peer-to-peer connection with the suspected member. If the maximum number of missed heartbeats is exceeded and peer-to-peer connection fails, the member is marked as a suspected failure. The default is 3.

`heartbeat-frequency-in-millis`
: Indicates the frequency (in milliseconds) at which a heartbeat is sent by each server instance to the cluster.

  The failure detection interval is the `max-missed-heartbeats` multiplied by the `heartbeat-frequency-in-millis`. Therefore, the combination of defaults, 3 multiplied by 2000 milliseconds, results in a failure detection interval of 6 seconds.

  Lowering the value of `heartbeat-frequency-in-millis` below the default would result in more frequent heartbeat messages being sent out from each member. This could potentially result in more heartbeat messages in the network than a system needs for triggering failure detection protocols. The effect of this varies depending on how quickly the deployment environment needs to have failure detection performed. That is, the (lower) number of retries with a lower heartbeat interval would make it quicker to detect failures.

  However, lowering this value could result in false positives because you could potentially detect a member as failed when, in fact, the member's heartbeat is reflecting the network load from other parts of the server. Conversely, a higher timeout interval results in fewer heartbeats in the system because the time interval between heartbeats is longer. As a result, failure detection would take a longer. In addition, a startup by a failed member during this time results in a new join notification but no failure notification, because failure detection and verification were not completed.

  The default is 2000.

`verify-failure-waittime-in-millis`
: Indicates the verify suspect protocol's timeout used by the health monitor. After a member is marked as suspect based on missed heartbeats and a failed peer–to–peer connection check,

the verify suspect protocol is activated and waits for the specified timeout to check for any further health state messages received in that time, and to see if a peer-to-peer connection can be made with the suspect member. If not, then the member is marked as failed and a failure notification is sent. The default is 1500.

`verify-failure-connect-timeout-in-millis`
Indicates the time it takes for the GMS to detect a hardware or network failure of a server instance. Be careful not to set this value too low. The smaller this timeout value is, the greater the chance of detecting false failures. That is, the instance has not failed but doesn't respond within the short window of time. The default is 10000.

The heartbeat frequency, maximum missed heartbeats, peer-to-peer connection-based failure detection, and the verify timeouts are all needed to ensure that failure detection is robust and reliable in GlassFish Server.

For the dotted names for each of these GMS configuration settings, see "Dotted Names for GMS Settings" on page 34. For the steps to specify these settings, see "To Preconfigure Nondefault GMS Configuration Settings" on page 35.

Dotted Names for GMS Settings

Below are sample `get(1)` subcommands to get all the GMS configuration settings (attributes associated with the referenced `mycfg` configuration) and GMS cluster settings (attributes and properties associated with a cluster named `mycluster`).

```
asadmin> get "configs.config.mycfg.group-management-service.*"
configs.config.mycfg.group-management-service.failure-detection.heartbeat-frequency-in-millis=2000
configs.config.mycfg.group-management-service.failure-detection.max-missed-heartbeats=3
configs.config.mycfg.group-management-service.failure-detection.verify-failure-connect-timeout-in-millis=10000
configs.config.mycfg.group-management-service.failure-detection.verify-failure-waittime-in-millis=1500
configs.config.mycfg.group-management-service.group-discovery-timeout-in-millis=5000
```

```
asadmin> get "clusters.cluster.mycluster.gms*"
clusters.cluster.mycluster.gms-bind-interface-address=${GMS-BIND-INTERFACE-ADDRESS-cluster2}
clusters.cluster.mycluster.gms-enabled=true
clusters.cluster.mycluster.gms-multicast-address=228.9.245.47
clusters.cluster.mycluster.gms-multicast-port=9833
```

```
asadmin> get "clusters.cluster.mycluster.property.*"
clusters.cluster.mycluster.property.GMS_LISTENER_PORT=${GMS_LISTENER_PORT-cluster2}
clusters.cluster.mycluster.property.GMS_MULTICAST_TIME_TO_LIVE=4
clusters.cluster.mycluster.property.GMS_LOOPBACK=false
clusters.cluster.mycluster.property.GMS_TCPSTARTPORT=9090
clusters.cluster.mycluster.property.GMS_TCPENDPORT=9200
```

For the steps to specify these settings, see "To Preconfigure Nondefault GMS Configuration Settings" on page 35 and "To Change GMS Settings After Cluster Creation" on page 36.

▼ To Preconfigure Nondefault GMS Configuration Settings

You can preconfigure GMS with values different than the defaults without requiring a restart of the DAS and the cluster.

1 Create a configuration using the `copy-config`(1) subcommand.

For example:

```
asadmin> copy-config mycfg
```

For more information, see "To Create a Named Configuration" on page 71.

2 Set the values for the new configuration's GMS configuration settings.

For example:

```
asadmin > set configs.config.mycfg.group-management-service.group-discovery-timeout-in-millis=8000
asadmin> set configs.config.mycfg.group-management-service.failure-detection.max-missed-heartbeats=5
```

For a complete list of the dotted names for these settings, see "Dotted Names for GMS Settings" on page 34.

3 Create the cluster so it uses the previously created configuration.

For example:

```
asadmin> create-cluster --config mycfg mycluster
```

You can also set GMS cluster settings during this step. For more information, see "To Create a Cluster" on page 40.

4 Create server instances for the cluster.

For example:

```
asadmin> create-instance --cluster mycluster instance01

asadmin> create-instance --cluster mycluster instance02
```

5 Start the cluster.

For example:

```
asadmin> start-cluster mycluster
```

See Also You can also view the full syntax and options of a subcommand by typing `asadmin help` *subcommand* at the command line.

▼ To Change GMS Settings After Cluster Creation

To avoid the need to restart the DAS and the cluster, configure GMS configuration settings before cluster creation as explained in "To Preconfigure Nondefault GMS Configuration Settings" on page 35.

To avoid the need to restart the DAS and the cluster, configure the GMS cluster settings during cluster creation as explained in "To Create a Cluster" on page 40.

Changing any GMS settings using the `set` subcommand after cluster creation requires a domain administration server (DAS) and cluster restart as explained here.

1. **Ensure that the DAS and cluster are running.**

   Remote subcommands require a running server.

2. **Use the `get`(1) subcommand to determine the settings to change.**

   For example:

```
asadmin> get "configs.config.mycfg.group-management-service.*"
configs.config.mycfg.group-management-service.failure-detection.heartbeat-frequency-in-millis=2000
configs.config.mycfg.group-management-service.failure-detection.max-missed-heartbeats=3
configs.config.mycfg.group-management-service.failure-detection.verify-failure-connect-timeout-in-millis=10000
configs.config.mycfg.group-management-service.failure-detection.verify-failure-waittime-in-millis=1500
configs.config.mycfg.group-management-service.group-discovery-timeout-in-millis=5000
```

   For a complete list of the dotted names for these settings, see "Dotted Names for GMS Settings" on page 34.

3. **Use the `set`(1) subcommand to change the settings.**

   For example:

```
asadmin> set configs.config.mycfg.group-management-service.group-discovery-timeout-in-millis=6000
```

4. **Use the `get` subcommand again to confirm that the changes were made.**

   For example:

```
asadmin> get configs.config.mycfg.group-management-service.group-discovery-timeout-in-millis
```

5. **Restart the DAS.**

   For example:

```
asadmin> stop-instance server

asadmin> start-instance server
```

6. **Restart the cluster.**

   For example:

```
asadmin> stop-cluster mycluster

asadmin> start-cluster mycluster
```

See Also You can also view the full syntax and options of a subcommand by typing `asadmin help` *subcommand* at the command line.

▼ To Check the Health of Instances in a Cluster

The `get-health` subcommand only works when GMS is enabled. This is the quickest way to evaluate the health of a cluster and to detect if cluster is properly operating; that is, all members of the cluster are running and visible to DAS.

If multicast is not enabled for the network, all instances could be running (as shown by the `list-instances`(1) subcommand), yet isolated from each other. The `get-health` subcommand does not show the instances if they are running but cannot discover each other due to multicast not being configured properly. See "To Validate that Multicast Transport Is Available for a Cluster" on page 37.

1 **Ensure that the DAS and cluster are running.**

Remote subcommands require a running server.

2 **Check whether server instances in a cluster are running by using the `get-health`(1) subcommand.**

Example 3–1 Checking the Health of Instances in a Cluster

This example checks the health of a cluster named `cluster1`.

```
asadmin> get-health cluster1
instance1 started since Wed Sep 29 16:32:46 EDT 2010
instance2 started since Wed Sep 29 16:32:45 EDT 2010
Command get-health executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help get-health` at the command line.

▼ To Validate that Multicast Transport Is Available for a Cluster

Before You Begin To test a specific multicast address, multicast port, or bind interface address, get this information beforehand using the `get` subcommand. Use the following subcommand to get the multicast address and port for a cluster named `c1`:

```
asadmin> get "clusters.cluster.c1.gms-multicast*"
clusters.cluster.c1.gms-multicast-address=228.9.174.162
clusters.cluster.c1.gms-multicast-port=5383
```

Use the following subcommand to get the bind interface address of a server instance named `i1`that belongs to a cluster named `c1`:

```
asadmin> get servers.server.i1.system-property.GMS-BIND-INTERFACE-ADDRESS-c1
servers.server.i1.system-property.GMS-BIND-INTERFACE-ADDRESS-c1.name=GMS-BIND-INTERFACE-ADDRESS-c1
servers.server.i1.system-property.GMS-BIND-INTERFACE-ADDRESS-c1.value=10.12.152.30
```

- **Check whether multicast transport is available for a cluster by using the `validate-multicast`(1) subcommand.**

Example 3–2 Validating that Multicast Transport Is Available for a Cluster

This example checks whether multicast transport is available for a cluster named `c1`.

```
asadmin> validate-multicast --multicastaddress=228.9.175.162 --multicastport=5383
Will use port 5,383
Will use address 228.9.175.162
Will use bind interface null
Will use wait period 2,000 (in milliseconds)

Listening for data...
Sending message with content "sr1" every 2,000 milliseconds
Received data from sr1 (loopback)
Exiting after 20 seconds. To change this timeout, use the --timeout command line option.
Command validate-multicast executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help get-health` at the command line.

Using the Multi-Homing Feature With GMS

Multi-homing enables GlassFish Server clusters to be used in an environment that uses multiple Network Interface Cards (NICs). A multi-homed host has multiple network connections, of which the connections may or may not be the same network. Multi-homing provides the following benefits:

- Provides redundant network connections within the same subnet. Having multiple NICs ensures that one or more network connections are available for communication.
- Supports communication across two or more different subnets. The DAS and all server instances in the same cluster must be on the same subnet for GMS communication, however.
- Binds to a specific IPv4 address and receives GMS messages in a system that has multiple IP addresses configured. The responses for GMS messages received on a particular interface will also go out through that interface.
- Supports separation of external and internal traffic.

Traffic Separation Using Multi-Homing

You can separate the internal traffic resulting from GMS from the external traffic. Traffic separation enables you plan a network better and augment certain parts of the network, as required.

Consider a simple cluster, `c1`, with three instances, `i101`, `i102`, and `i103`. Each instance runs on a different machine. In order to separate the traffic, the multi-homed machine should have at least two IP addresses belonging to different networks. The first IP as the external IP and the second one as internal IP. The objective is to expose the external IP to user requests, so that all the traffic from the user requests would be through them. The internal IP is used only by the cluster instances for internal communication through GMS. The following procedure describes how to set up traffic separation.

To configure multi-homed machines for GMS without traffic separation, skip the steps or commands that configure the `EXTERNAL-ADDR` system property, but perform the others.

To avoid having to restart the DAS or cluster, perform the following steps in the specified order.

▼ To Set Up Traffic Separation

1 **Create the system properties `EXTERNAL-ADDR` and `GMS-BIND-INTERFACE-ADDRESS-c1` for the DAS.**

- `asadmin create-system-properties --target server EXTERNAL-ADDR=192.155.35.4`
- `asadmin create-system-properties --target server GMS-BIND-INTERFACE-ADDRESS-c1=10.12.152.20`

2 **Create the cluster with the default settings.**

Use the following command:

```
asadmin create-cluster c1
```

A reference to a system property for GMS traffic is already set up by default in the `gms-bind-interface-address` cluster setting. The default value of this setting is `${GMS-BIND-INTERFACE-ADDRESS-`*cluster-name*`}`.

3 **When creating the clustered instances, configure the external and GMS IP addresses.**

Use the following commands:

- `asadmin create-instance --node localhost --cluster c1 --systemproperties EXTERNAL-ADDR=192.155.35.5:GMS-BIND-INTERFACE-ADDRESS-c1=10.12.152.30 i101`
- `asadmin create-instance --node localhost --cluster c1 --systemproperties EXTERNAL-ADDR=192.155.35.6:GMS-BIND-INTERFACE-ADDRESS-c1=10.12.152.40 i102`
- `asadmin create-instance --node localhost --cluster c1 --systemproperties EXTERNAL-ADDR=192.155.35.7:GMS-BIND-INTERFACE-ADDRESS-c1=10.12.152.50 i103`

4 Set the address attribute of HTTP listeners to refer to the `EXTERNAL-ADDR` system properties.

Use the following commands:

```
asadmin set c1-config.network-config.network-listeners.network-listener.http-1.address=\${EXTERNAL-ADDR}
asadmin set c1-config.network-config.network-listeners.network-listener.http-2.address=\${EXTERNAL-ADDR}
```

Creating, Listing, and Deleting Clusters

GlassFish Server enables you to create clusters, obtain information about clusters, and delete clusters that are no longer required.

The following topics are addressed here:

- "To Create a Cluster" on page 40
- "To List All Clusters in a Domain" on page 43
- "To Delete a Cluster" on page 44

▼ To Create a Cluster

Use the `create-cluster` subcommand in remote mode to create a cluster.

To ensure that the GMS can detect changes in cluster membership, a cluster's GMS settings must be configured correctly. To avoid the need to restart the DAS and the cluster, configure a cluster's GMS settings when you create the cluster. If you change GMS settings for an existing cluster, the DAS and the cluster must be restarted to apply the changes.

When you create a cluster, GlassFish Server automatically creates a Message Queue cluster for the GlassFish Server cluster. For more information about Message Queue clusters, see "Using MQ Clusters with GlassFish Server" on page 155.

Before You Begin If the cluster is to reference an existing named configuration, ensure that the configuration exists. For more information, see "To Create a Named Configuration" on page 71. If you are using a named configuration to preconfigure GMS settings, ensure that these settings have the required values in the named configuration. For more information, see "To Preconfigure Nondefault GMS Configuration Settings" on page 35.

If you are configuring the cluster's GMS settings when you create the cluster, ensure that you have the following information:

- The address on which GMS listens for group events
- The port number of the communication port on which GMS listens for group events
- The maximum number of iterations or transmissions that a multicast message for GMS events can experience before the message is discarded
- The lowest port number in the range of ports from which GMS selects a TCP port on which to listen

- The highest port number in the range of ports from which GMS selects a TCP port on which to listen

If the DAS is running on a multihome host, ensure that you have the Internet Protocol (IP) address of the network interface on the DAS host to which GMS binds.

1 **Ensure that the DAS is running.**

Remote subcommands require a running server.

2 **Run the `create-cluster` subcommand.**

Note – Only the options that are required to complete this task are provided in this step. For information about all the options for configuring the cluster, see the `create-cluster`(1) help page.

```
asadmin> create-cluster  --config configuration
--multicastaddress multicast-address --multicastport multicast-port
--properties GMS_MULTICAST_TIME_TO_LIVE=max-iterations:
GMS_TCPSTARTPORT=start-port:GMS_TCPENDPORT=end-port  cluster-name
```

configuration
: An existing named configuration that the cluster is to reference.

multicast-address
: The address on which GMS listens for group events.

multicast-port
: The port number of the communication port on which GMS listens for group events.

max-iterations
: The maximum number of iterations or transmissions that a multicast message for GMS events can experience before the message is discarded.

start-port
: The lowest port number in the range of ports from which GMS selects a TCP port on which to listen. The default is 9090.

end-port
: The highest port number in the range of ports from which GMS selects a TCP port on which to listen. The default is 9200.

cluster-name
: Your choice of name for the cluster that you are creating.

3 If necessary, create a system property to represent the IP address of the network interface on the DAS host to which GMS binds.

This step is necessary only if the DAS is running on a multihome host.

```
asadmin> create-system-properties
GMS-BIND-INTERFACE-ADDRESS-cluster-name=das-bind-address
```

cluster-name
The name that you assigned to the cluster in Step 2.

das-bind-address
The IP address of the network interface on the DAS host to which GMS binds.

Example 3–3 Creating a Cluster

This example creates a cluster that is named `ltscluster` for which port 1169 is to be used for secure IIOP connections. Because the `--config` option is not specified, the cluster references a copy of the named configuration `default-config` that is named `ltscluster-config`.

```
asadmin> create-cluster
--systemproperties IIOP_SSL_LISTENER_PORT=1169
ltscluster
Command create-cluster executed successfully.
```

Example 3–4 Creating a Cluster and Setting GMS Options

This example creates a cluster that is named `pmdcluster`, which references the existing configuration `clusterpresets` and for which the cluster's GMS settings are configured as follows:

- GMS listens for group events on address 228.9.3.1 and port 2048.
- A multicast message for GMS events is discarded after 3 iterations or transmissions.
- GMS selects a TCP port on which to listen from ports in the range 10000–10100.

```
asadmin> create-cluster --config clusterpresets
--multicastaddress 228.9.3.1 --multicastport 2048
--properties GMS_MULTICAST_TIME_TO_LIVE=3:
GMS_TCPSTARTPORT=10000:GMS_TCPENDPORT=10100 pmdcluster
Command create-cluster executed successfully.
```

Next Steps After creating a cluster, you can add GlassFish Server instances to the cluster as explained in the following sections:

- "To Create an Instance Centrally" on page 48
- "To Create an Instance Locally" on page 58

See Also

- "To Create a Named Configuration" on page 71
- "To Preconfigure Nondefault GMS Configuration Settings" on page 35

- "Using MQ Clusters with GlassFish Server" on page 155
- `create-cluster`(1)
- `create-system-properties`(1)

You can also view the full syntax and options of the subcommands by typing the following commands at the command line:

- `asadmin help create-cluster`
- `asadmin help create-system-properties`

▼ To List All Clusters in a Domain

Use the `list-clusters` subcommand in remote mode to obtain information about existing clusters in a domain.

1 **Ensure that the DAS is running.**

Remote subcommands require a running server.

2 **Run the `list-clusters`(1) subcommand.**

```
asadmin> list-clusters
```

Example 3–5 Listing All Clusters in a Domain

This example lists all clusters in the current domain.

```
asadmin> list-clusters
pmdclust not running
ymlclust not running
Command list-clusters executed successfully.
```

Example 3–6 Listing All Clusters That Are Associated With a Node

This example lists the clusters that contain an instance that resides on the node `sj01`.

```
asadmin> list-clusters sj01
ymlclust not running
Command list-clusters executed successfully.
```

See Also `list-clusters`(1)

You can also view the full syntax and options of the subcommand by typing `asadmin help list-clusters` at the command line.

▼ To Delete a Cluster

Use the `delete-cluster` subcommand in remote mode to remove a cluster from the DAS configuration.

If the cluster's named configuration was created automatically for the cluster and no other clusters or unclustered instances refer to the configuration, the configuration is deleted when the cluster is deleted.

Before You Begin Ensure that following prerequisites are met:

- The cluster that you are deleting is stopped. For information about how to stop a cluster, see "To Stop a Cluster" on page 54.
- The cluster that you are deleting contains no GlassFish Server instances. For information about how to remove instances from a cluster, see the following sections:
  - "To Delete an Instance Centrally" on page 52
  - "To Delete an Instance Locally" on page 61

1 Ensure that the DAS is running.

Remote subcommands require a running server.

2 Confirm that the cluster is stopped.

```
asadmin> list-clusters cluster-name
```

cluster-name
The name of the cluster that you are deleting.

3 Confirm that the cluster contains no instances.

```
asadmin> list-instances cluster-name
```

cluster-name
The name of the cluster that you are deleting.

4 Run the `delete-cluster`(1) subcommand.

```
asadmin> delete-cluster cluster-name
```

cluster-name
The name of the cluster that you are deleting.

Example 3–7 Deleting a Cluster

This example confirms that the cluster `adccluster` is stopped and contains no instances and deletes the cluster `adccluster`.

```
asadmin> list-clusters adccluster
adccluster not running
Command list-clusters executed successfully.
asadmin> list-instances adccluster
Nothing to list.
Command list-instances executed successfully.
asadmin> delete-cluster adccluster
Command delete-cluster executed successfully.
```

See Also

- "To Stop a Cluster" on page 54
- "To Delete an Instance Centrally" on page 52
- "To Delete an Instance Locally" on page 61
- `delete-cluster`(1)
- `list-clusters`(1)
- `list-instances`(1)

You can also view the full syntax and options of the subcommands by typing the following commands at the command line:

- `asadmin help delete-cluster`
- `asadmin help list-clusters`
- `asadmin help list-instances`

GlassFish Server Open Source Edition 3.1 Review Draft 1   Composed January 20, 2011