BEA WebLogic Server®

Automatic and Manual Service-level Migration

Version 10.3 Technical Preview
Revised: March 2007

Service-Level Migration

New in WebLogic Server 10.3: Automatic Migration of Messaging/JMS-Related Services

The WebLogic Server migration framework allows an administrator to configure migratable targets so that certain Messaging/JMS-related services are automatically migrated from an unhealthy hosting server to an another available server with the help of server health monitoring capabilities. This capability improves the availability of migratable Messaging/JMS-related services in a cluster because those services can be quickly restarted on a redundant server should the host server fail.

WebLogic Server supports automatic service-level migration for the following Messaging/JMS-related services:

- JMS Server
- SAF Service
- Path Service
- Persistent Store

When a migratable pinned Messaging/JMS-related service fails or become unavailable for any reason (for example, because of a bug in the service code, server failure, or network failure), it is deactivated at its current location and activated on a new server.

The following sections describe the service-level migration mechanisms supported by WebLogic Server:

- Understanding the Service-Level Migration Framework
- Pre-Migration Requirements
- Roadmap for Configuring Automatic Migration of Messaging/JMS-Related Services
- Roadmap for Configuring Manual Migration of Messaging/JMS-Related Services
- Roadmap for Configuring Automatic Migration of the JTA Transaction Recovery Service
- Manual Migration of the JTA Transaction Recovery Service
- Automatic Migration of User-Defined Singleton Services

These sections focus on the migration of failed services. WebLogic Server also supports whole server-level migration, where a migratable server instance, and all of its services, is migrated to a different physical machine upon failure. For information on failed server migration, See Whole Server Migration in *Using WebLogic Server Clusters*.

WebLogic Server also supports replication and failover at the application level. For more information, see Failover and Replication in a Cluster in *Using WebLogic Server Clusters*.

Understanding the Service-Level Migration Framework

In a WebLogic Server cluster, most subsystem services are hosted homogeneously on all server instances in the cluster, enabling transparent failover from one server to another. In contrast, *pinned services*, such as Messaging/JMS-related services, the JTA Transaction Recovery Service, and user-defined singleton services are hosted on individual server instances within a cluster—for these services, the WebLogic Server migration framework supports failure recovery with *service migration*, as opposed to failover. See "Migratable Services" on page -3.

Service-level migration in WebLogic Server is the process of moving the pinned services from one instance server instance to a different available server instance within the cluster. Service migration is controlled by logical *migratable target*, which serves as a grouping of services that is hosted on only one physical server in a cluster. You can select a migratable target in place of a server or cluster when targeting certain pinned services. High availability is achieved by migrating a migratable target from one clustered server to another when a problem occurs on the original server. You can also manually migrate a migratable target for scheduled maintenance or you can configure the migratable target for automatic migration. See "Understanding Migratable Targets In a Cluster" on page -5.

The migration framework provides tools and infrastructure for configuring and migrating targets, and, in the case of automatic service migration, it leverages WebLogic Server's health monitoring subsystem to monitor the health of services hosted by a migratable target. See "Migration Processing Tools" on page -11 and "Automatic Service Migration Infrastructure" on page -12. For definitions of the terms that apply to server and service migration, see "Migration Terminology" on page 7-2.

Migratable Services

WebLogic Server supports service-level migration for JMS-related services, the JTA Transaction Recovery Service, and user-defined singleton services. These are referred to as *migratable services*, because you can move them from one server to another within a cluster. The following migratable services can be configured for automatic or manual migration:

Messaging/JMS-related Services

JMS services are singleton services, and, therefore, are not active on all server instances in a cluster. Instead, they are pinned to a single server in the cluster to preserve data consistency. To ensure that singleton JMS services do not introduce a single point of failure for dependent applications in the cluster, WebLogic Server can be configured to automatically or manually migrate them to any server instance in the migratable target list.

- JMS Server – management containers for the queues and topics in JMS modules that are targeted to them. See JMS Server Configuration in *Configuring and Managing WebLogic JMS*.
- Store-and-Forward (SAF) Service – store-and-forward messages between local sending and remote receiving endpoints, even when the remote endpoint is not available at the moment the messages are sent. Only sending SAF agents configured for JMS SAF (sending capability only) are migratable. See *Configuring and Managing WebLogic Store-and-Forward*.
- Path Service – a persistent map that can be used to store the mapping of a group of messages in a JMS Message Unit-of-Order to a messaging resource in a cluster. It provides a way to enforce ordering by pinning messages to a member of a cluster hosting servlets, distributed queue members, or Store-and-Forward agents. One path service is configured per cluster. See Using the WebLogic Path Service in *Configuring and Managing WebLogic JMS*.
- Custom Persistent Store – a user-defined, disk-based file store or JDBC-accessible database for storing subsystem data, such as persistent JMS messages or store-and-forward messages. See Using the WebLogic Persistent Store in *Configuring WebLogic Server Environments*.

JTA Transaction Recovery Service

The Transaction Recovery Service automatically attempts to recover transactions on system startup by parsing all transaction log records for incomplete transactions and completing them. For detailed information, see Transaction Recovery After a Server Fails in *Programming WebLogic JTA*.

User-defined Singleton Services

Within an application, you can define a singleton service that can be used to perform tasks that you want to be executed on only one member of a cluster at any give time. See "Automatic Migration of User-Defined Singleton Services" on page -32.

Understanding Migratable Targets In a Cluster

You can configure JMS and JTA services for high availability by using migratable targets. A migratable target is a special target that can migrate from one server in a cluster to another. As such, a migratable target provides a way to group migratable services that should move together. When the migratable target is migrated, all services hosted by that target are migrated.

In order to configure a migratable JMS service for migration, it must be deployed to a migratable target. A migratable target specifies a set of servers that can host a target, and can optionally specify a user-preferred host for the services and an ordered list of candidate backup servers should the preferred server fail. Only one of these servers can host the migratable target at any one time.

Once a service is configured to use a migratable target, then the service is independent from the server member that is currently hosting it. For example, if a JMS server with a deployed JMS queue is configured to use a migratable target, then the queue is independent of when a specific server member is available. In other words, the queue is always available when the migratable target is hosted by any server in the cluster.

An administrator can manually migrate pinned migratable services from one server instance to another in the cluster, either in response to a server failure or as part of regularly scheduled maintenance. If you do not configure a migratable target in the cluster, migratable services can be migrated to any WebLogic Server instance in the cluster. See the "Roadmap for Configuring Manual Migration of Messaging/JMS-Related Services" on page -23.

Policies for Manual and Automatic Service Migration

A migratable target provides migration policies that define whether the hosted services will be manually migrated (the system default) or automatically migrated from an unhealthy hosting server to a healthy active server with the help of the Health Monitoring subsystem. There are two types of automatic service migration policies, as described in the following sections.

Manual Migration

When a migratable target uses the `manual` policy (the system default), an administrator can manually migrate pinned migratable services from one server instance to another in the cluster, either in response to a server failure or as part of regularly scheduled maintenance.

See the "Roadmap for Configuring Manual Migration of Messaging/JMS-Related Services" on page -23.

Exactly-Once

This policy indicates that if at least one Managed Server in the candidate list is running, then the service will be active somewhere in the cluster. It is important to note that this value can lead to target grouping. For example, if you have five `exactly-once` migratable targets and only boot one Managed Server in the cluster, then all five targets will be activated on that server.

Tip: As a best practice, a migratable target hosting a path service should always be set to `exactly-once`, so if its hosting server fails or is shut down, then the path service will automatically migrate to another server and so will always be active in the cluster.

Example use-case for JMS servers:
A domain has a cluster of three Managed Servers, with one JMS server deployed on a member server in the cluster. Applications deployed to the cluster send messages to the queues targeted to the JMS server. MDBs in another domain drain the queues associated with the JMS server. The MDBs only want to drain from one set of queues, not from many instances of the same queue. In other words, this environment uses clustering for scalability, load balancing, and failover for its *applications*, but not for its JMS server. Therefore, this environment would benefit from the automatic migration of the JMS server as an `exactly-once` service to an available cluster member.

See the "Roadmap for Configuring Automatic Migration of Messaging/JMS-Related Services" on page -18.

Failure-Recovery

This policy indicates that the service will only start if its User Preferred Server (UPS) is started. If the UPS is shutdown gracefully, this service will not be migrated anywhere. However, if the UPS crashes, is forcibly shutdown, or is otherwise made unavailable, then this service will be migrated to another candidate server. In addition, if such a migrated `failure-recovery` service is gracefully shutdown on a non-UPS server, it should be migrated to another candidate server.

Example use-case for JMS servers:
A domain has a cluster of three Managed Servers, with a JMS server on each member server and a distributed queue member on each JMS server. There is also an MDB targeted to the cluster that drains from the distributed queue member on the local server member. In other words, this environment uses clustering for overall scalability, load balancing, and failover. Therefore, this environment would benefit from the automatic migration of a JMS server as an `failure-recovery` service to a UPS member.

Caution: If a server is also configured to use the automatic *whole-server* migration framework, which will shut down the server when its expired lease cannot be renewed, then any `failure-recovery` services also configured on that server will not automatically migrate no matter how the server is shutdown (e.g., killed vs.graceful shutdown). For more information, see "Automatic Server Migration" on page 7-6.

See the "Roadmap for Configuring Automatic Migration of Messaging/JMS-Related Services" on page -18.

Options For Attempting to Restart Failed Services Before Migrating

A migratable target provides options to attempt to deactivate and reactivate a failed service, instead of migrating the service. See "In-Place Restarting of Failed Migratable Services" on page -14.

For more information about the default values for all migratable target options, see `MigratableTargetMBean` in the *WebLogic Server MBean Reference*.

User-Preferred Servers and Candidate Servers

When deploying a JMS service to the migratable target, you can select a the user-preferred server (UPS) target to host the service. When configuring a migratable target, you can also specify constrained candidate servers (CCS) that can potentially host the service should the user-preferred server fail. If the migratable target does not specify a constrained candidate server, the JMS server can be migrated to any available server in the cluster.

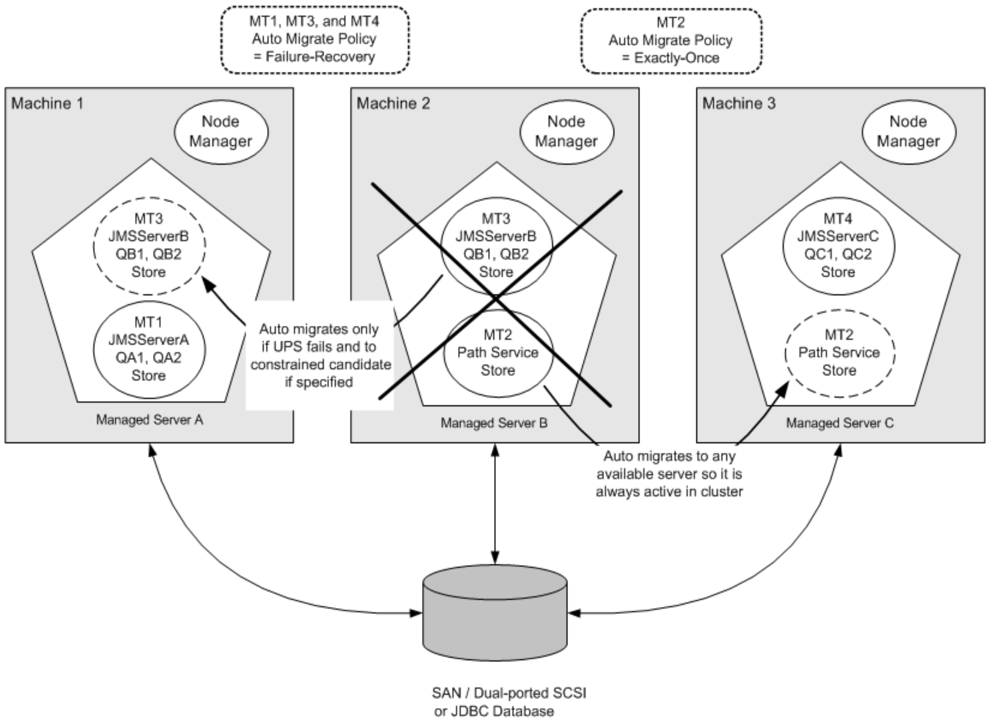
WebLogic Server enables you to create separate migratable targets for JMS services. This allows you to always keep each service running on a different server in the cluster, if necessary. Conversely, you can configure the same selection of servers as the constrained candidate servers for both JTA and JMS, to ensure that the services remain co-located on the same server in the cluster.

Example Migratable Targets In a Cluster

The following figure shows a cluster of three Managed Servers, all hosting migratable targets. Server A is hosting a migratable target (MT1) for JMS server A (with two queues) and a custom store; Server B is hosting MT2 for a path service and a custom store and is also hosting MT3 for JMS server B (with two queues) and a custom store; Server C is hosting MT4 for JMS server C (with two queues) and a custom store.

All the migratable targets are configured to be automatically migrated, with the MT1, MT3, and MT4 targets using the `failure-recovery` policy, and the MT2 target using the `exactly-once` policy.

Figure 7-4 Migratable Targets In a Cluster


MT1, MT3, and MT4 Auto Migrate Policy = Failure-Recovery
MT2 Auto Migrate Policy = Exactly-Once
Machine 1
Node Manager
MT3 JMSServerB QB1, QB2 Store
MT1 JMSServerA QA1, QA2 Store
Managed Server A
Machine 2
Node Manager
MT3 JMSServerB QB1, QB2 Store
MT2 Path Service Store
Managed Server B
Machine 3
Node Manager
MT4 JMSServerC QC1, QC2 Store
MT2 Path Service Store
Managed Server C
Auto migrates only if UPS fails and to constrained candidate if specified
Auto migrates to any available server so it is always active in cluster
SAN / Dual-ported SCSI or JDBC Database

In the above example, the MT2 `exactly-once` target will automatically start the path service and store on any running Managed Server in the candidate list. This way, if the hosting server should fail, it guarantees that the services will always be active somewhere in the cluster, even if the targets user preferred server (UPS) is shut down gracefully. However, as described in "Policies for Manual and Automatic Service Migration" on page -5, this policy can also lead to target grouping with multiple JMS services being hosted a single server.

Whereas, the MT1, MT3, and MT4 `failure-recovery` targets will only automatically start the JMS server and store services on its UPS, and if the UPS is shutdown gracefully, the pinned services will not be migrated anywhere. However, if the UPS crashes or is forcibly shutdown, then the services will be migrated to another preferred candidate server.

Targeting Rules for JMS Servers

When not using migratable targets, a JMS server can be targeted to a specific cluster member and can use either the default file or a custom store. However, when targeted to a migratable target, a JMS server must use a custom persistent store, and must be targeted to the same migratable target used by the custom store. A JMS server, SAF agent, and custom store can share a migratable target. See "Custom Store Availability for JMS Services" on page -15.

Targeting Rules for SAF Agents

When not using migratable targets, a SAF agent can be targeted to an entire cluster or a list of multiple servers in a cluster, with the requirement that the SAF agent and each server in the cluster must use the default persistent store. However, when targeted to a migratable target, a SAF agent can only be targeted to that migratable target. It must also use a custom persistent store, and, like a JMS server, must targeted to the same migratable target used by the custom store. A SAF agent, JMS server, and custom store can share a migratable target. See "Special Considerations When Targeting SAF Agents or Path Service" on page -21.

In addition, consider the following topics when targeting SAF agents to migratable targets.

Re-targeting SAF Agents to Migratable Targets

To preserve SAF message consistency, WebLogic Server prevents you from retargeting an existing SAF agent to a migratable target. Instead, you must delete the existing SAF agent and configure a new one with the same values and target it to a migratable target.

Targeting Migratable SAF Agents For Increased Message Throughput

When not using migratable targets, a SAF agent can be targeted to an entire cluster or multiple servers in a cluster for increased message throughput. However, When a SAF agent is targeted to a migratable target, it cannot be targeted to any other servers in the cluster, including an entire cluster. Therefore, if you want to increase throughput by importing a JMS destination to multiple

SAF agents on separate servers in a cluster, then you should create migratable targets for each server in the cluster, and then create separate SAF agents that are targeted individually to each migratable target.

Targeting SAF Agents For Consistent Quality-of-Service

A Weblogic administrator has the freedom to configure and deploy multiple SAF agents in the same cluster or on the same server. As such, there could be situations where the same server has both migratable SAF agents and non-migratable ones. For such cases, the behavior of a JMS client application may vary depending on which SAF agent handles the messages.

For example, an imported destination can be deployed to multiple SAF agents, and messages sent to the imported destination will be load-balanced among all SAF agents. If the list of the SAF agents contains non-migratable agents, the JMS client application may have a limited sense of HA (high availability). Therefore, a recommended best practice is to deploy an imported destination to one or more SAF agents that provide the same level of HA functionality. In other words, to get consistent forwarding quality and behavior, you should target the imported destination to a set of SAF agents that are all targeted to migratable targets or are all targeted to non-migratable targets

Targeting Rules for Path Service

When not using migratable targets, a path service is targeted to single member of a cluster, and can use either the default file or a custom store. However, when targeted to a migratable target, a path service cannot use the default store, so a custom store must be configured and targeted to the same migratable target. As an additional best practice, the path service and its custom store should be the only users of that migratable target. Whereas, a JMS server, SAF agent, and custom store can share a migratable target.

Special Considerations For Targeting a Path Service

When the path service for a cluster is targeted to a migratable target, as a best practice, the path service and its custom store should be the only users of that migratable target.

When a path service is targeted to a migratable target its provides enhanced storage of message unit-of-order (UOO) information for JMS distributed destinations, since the UOO information will be based on the entire migratable target instead of being based only on the server instance hosting the distributed destinations member.

Targeting Rules for Custom Stores

As mentioned previously, all Messaging/JMS-related services require a custom persistent store. that is targeted to the same migratable targets as the JMS services. When a custom store is targeted to a migratable target, the store's `<directory>` parameter must be configured so that the store directory is accessible from all candidate server members in the migratable target.

See "Custom Store Availability for JMS Services" on page -15.

Migratable Targets For the JTA Transaction Recovery Service

For JTA, migratable target configuration is not necessary for automatic or manual migration because a migratable target is automatically defined for JTA at the server level. The default migration policy for JTA is manual, but when configured for automatic migration, the JTA policy is internally set to `failure-recovery`. This means that Transaction Recovery Service will only start if its user-preferred server (UPS) is started. If an administrator shuts down the UPS either gracefully or forcefully, this service will not be migrated anywhere. However, if the UPS shuts down due to an internal error, then this service will be migrated to another candidate server.

Migration Processing Tools

WebLogic Server migration framework provides infrastructure and facilities to perform the manual or automatic migration of Messaging/JMS-related services and the JTA Transaction Recovery Service. By default, an administrator has to manually execute the process in order to successfully migrate the services from one server instance to another server instance. However, these services can also be easily configured to automatically migrate in response to a server failure.

Administration Console

An administrator can use the WebLogic Administration Console to configure and/or perform the migration process.

For more information, see the following topics in the *Administration Console Help*:

- Configure JMS service migration
- Configure JTA Transaction Recovery Service migration

WebLogic Scripting Tool

An administrator can use the WebLogic Scripting Tool (WLST) command-line interface utility to manage the life cycle of a server instance, including configuring and/or performing the migration process.

For more information, refer to the Life Cycle Commands in *WebLogic Scripting Tool*.

Automatic Service Migration Infrastructure

The service migration framework depends on the following components to monitor server health issues and, if necessary, automatically migrate the pinned services to a healthy server.

Leasing for Migratable Services

Leasing is the process WebLogic Server uses to manage services that are required to run on only one member of a cluster at a time. Leasing ensures exclusive ownership of a cluster-wide entity. Within a cluster, there is a single owner of a lease. Additionally, leases can failover in case of server or cluster failure. This helps to avoid having a single point of failure. See "Leasing" on page 7-3.

Using the Automatic Migration option requires setting a cluster's **Migration Basis** policy to either **Database** or **Consensus** leasing, as follows:

Database Leasing

If you are using a database to manage leasing information, configure the database for server migration according to the procedures outlined in "High-availability Database Leasing" on page 7-5.

Setting **Migration Basis** to **Database** leasing requires that the **Data Source For Automatic Migration** option is set with a valid JDBC System Resource. It implies that there is a table created on that resource that the Managed Servers will use for leasing. For more information on creating a JDBC data source, see "Configuring JDBC Data Sources" in *Configuring and Managing WebLogic JDBC*.

Consensus Leasing

Setting **Migration Basis** to **Consensus** leasing means that the member servers maintain leasing information in-memory, which removes the requirement of having a high-availability database to use leasing. This version of leasing requires that you use Node Manager to control servers within the cluster. It also requires that all servers that are migratable, or which could host a migratable target, must have a Node Manager associated with them. The Node Manager is required to get health monitoring information about the member servers involved. See "Non-database Consensus Leasing" on page 7-6.

Node Manager

When using automatic service migration, the Node Manager is required to get health monitoring information about the member servers involved, as follows:

- Consensus leasing – Node Manager must be running on every machine hosting managed servers within the cluster.
- Database leasing – Node Manager must be running on every machine hosting managed servers within the cluster only if pre/post-migration scripts are defined. If pre/post-migrations are not defined, then Node manager is not required.

For general information on configuring Node Manager, see Using Node Manager to Control Servers.

Administration Server Not Required When Migrating Services

To eliminate a single point of failure during migration, the automatic service migration of migratable services is *not* dependent on the availability of the Administration Server at the time of migration.

Service Health Monitoring

To accommodate service migration requests, the migratable target performs basic health monitoring on migratable services deployed on it that implement a Health Monitoring Interface. The advantage of having a migratable target do this job is that it is guaranteed to be local. Plus, the migratable target has a direct communication channel to the leasing system, and can request that the lease be released (thus triggering a migration) when bad health is detected.

How Health Monitoring of the JTA Transaction Recovery Service Triggers Automatic Migration

When JTA has automatic migration enabled, the server defaults to shutting down if the JTA subsystem reports itself as unhealthy (`FAILED`). For example, if any IO error occurs when accessing the TLOG, then JTA health state will change to `FAILED`.

When the primary server fails, the migratable service framework automatically migrates the Transaction Recovery Service to a backup server. The automatic service migration framework selects a backup server from the configured candidate servers. If a backup server fails before completing the transaction recovery actions, and then is restarted, the Transaction Recovery Service will eventually be migrated to another server in the cluster (either the primary server will reclaim it or the migration framework will notice that the backup server's lease has expired).

After successful migration, if the backup server is shut down normally, then when the backup server is rebooted, the Transaction Recovery Service will again be activated on the backup server. This is consistent with manual service migration. As with manual service migration, the Transaction Recovery Service service cannot be migrated from a running primary server.

How Health Monitoring of Messaging/JMS-Related Services Triggers Automatic Migration

When the Messaging/JMS-related services have automatic migration enabled,

- **JMS Server** – Maintains its run-time health state and registers/updates its health to the Health Monitoring subsystem. When a service the JMS server depends upon, such as its targeted persistent store, reports the `FAILED` health state, it is detected by the migration framework, and the migration process takes place based on the migratable target's configured automatic migration policy. Typically, the migration framework deactivates the JMS server, and other users of the migratable target, on the current *user-preferred* server and onto a healthy available server from the constrained candidate server list.
- **SAF Service** – The health state of the SAF service comes from its configured SAF agents. If the SAF service detects an unhealthy state, the whole SAF agent instance will be reported as unhealthy. The SAF agent has the same health monitoring capabilities as a JMS server. Typically, the migration framework deactivates the SAF agent on the current user-preferred server and onto a healthy available server from the constrained candidate server list.
- **Path Service** – The path service itself will not change its health state, but instead depends on the server and its custom store to trigger migration.
- **Persistent Store** – Registers its health to the health monitoring subsystem. Any errors reported by the I/O layer such that if the persistent store cannot continue with read/write, and it needs to be restarted before it can guarantee data consistency, then the store's health is marked as `FAILED` and reported as `FAILED` to the health monitoring subsystem. This is detected by the automatic migration framework and triggers the auto-migration of the store and the subsystem services that are depending on that store from the current user- preferred server onto a healthy available server from the constrained candidate server list.

In-Place Restarting of Failed Migratable Services

Some migratable services, such as JMS, have the unique requirement that sometimes it is beneficial for the service to be restarted in place, instead of migrated. Therefore, migratable targets provide `restart-in-place` options to attempt to deactivate and reactivate a failed service, instead of migrating the service.

The migration framework only attempts to restart a service if the server's health is satisfactory (i.e., in a `RUNNING` state). If the server is not healthy for whatever reason, the framework immediately proceeds to the migration stage, skipping all in-place restarts.

The cluster's Singleton Monitor checks for the `RestartOnFailure` value in the service's `MigratableTargetMBean`. If it the value is `false`, then the service is migrated. If the value is `true`, then the migration framework attempts to deactivate/activate in place. If the reactivation fails, the migration framework pauses for the user-specified `SecondsBetweenRestarts` seconds. This is repeated for the specified `NumberOfRestartAttempts` attempts. If all restart attempts fail, then the service is migrated to a healthy server member.

Migrating a Service From an Unavailable Server

There are special considerations when you migrate a service from a server instance that has crashed or is unavailable to the Administration Server. If the Administration Server cannot reach the previously active host of the service at the time you perform the migration, that Managed Server's local configuration information (i.e., migratable target) will not be updated to reflect that it is no longer the active host for the service. In this situation, you must purge the unreachable Managed Server's local configuration cache before starting it again. This prevents the previous active host from hosting a service that has been migrated to another Managed Server.

Pre-Migration Requirements

WebLogic Server imposes certain constraints and prerequisites in terms of the service configuration in order to support service migration. These constraints are service specific and also depend your enterprise application architecture.

Custom Store Availability for JMS Services

Migratable Messaging/JMS-related services cannot use the default persistent store, so you must configure a custom store and target it to the same migratable target as the JMS server or SAF agent. (As a best practice, a path service should use its own custom store and migratable target).

The custom store must either be:

- Accessible from all candidate server member for both file-based and JDBC-accessible stores. Therefore, the store's `<directory>` parameter must be configured so that the store directory is accessible from all candidate server members in the migratable target.

   - If the application uses file-based persistence (file store), it is recommended to have either a SAN (Storage Area Network) or a dual-ported SCSI disk.
   - If the application uses JDBC-based persistence (JDBC store), then the JDBC connection information for that database instance, such as data source and connection pool, has to be available from all candidate servers.

- Migrated to a backup server target by pre-migration/post-migration scripts in the `BEA_HOME/user_projects/domains/mydomain/bin/service_migration` directory, where *`mydomain`* is a domain-specific directory, with the same name as the domain.

  Note: Basic directions for creating pre/post-migration scripts are provided in the `readme.txt` file in this directory.

  In some cases, scripts may be needed to dismount the disk from the previous server and mount it on the backup server. These scripts are configured on the Node Manager, using the `PreScript()` and `PostScript()` methods in the `MigratableTargetMBean`, or using the Administration Console. In other cases, a script may be needed to move (not copy) a custom file store directory to the backup server. The old configured file store directory should not be left for the next time the migratable target is hosted by the old server; therefore, the WebLogic administrator should delete or move the files to another directory.

Default File Store Availability for JTA

To migrate the JTA Transaction Recovery Service from a failed server in a cluster to another server (backup server) in the same cluster, the backup server must have access to the transaction log (TLOG) records from the failed server. Transaction log records are stored in the default persistent store for the server.

If you plan to use service migration in the event of a failure, you must configure the default persistent store so that it stores records in a shared storage system that is accessible to any potential machine to which a failed migratable server might be migrated. For highest reliability, use a shared storage solution that is itself highly available—for example, a storage area network (SAN) or a dual-ported disk. In addition, only JTA and other *non-migratable* services can share the same default store.

Optionally, you may also want to use pre/post-migration scripts to perform any unmounting and mounting of shared storage, as needed. Basic directions for creating pre/post-migration scripts are provided in a `readme.txt` file in the `BEA_HOME/user_projects/domains/mydomain/bin/service_migration` directory, where *`mydomain`* is a domain-specific directory, with the same name as the domain.

Server State and Manual Service Migration

For automatic migration, when the current (source) server fails, the migration framework will automatically migrate the Transaction Recovery Service to a target backup server.

For manual migration, you cannot migrate the Transaction Recovery Service to a backup server from a running server. You must stop the server before migrating the Transactions Recovery Service.

Table 7-1 Server Running State and Manual Migration Support

Server State Information		Migration Allowed?	
Current Server	**Backup Server**	**Messaging**	**JTA**
Running	Running	Yes	No
Running	Standby	Yes	No
Running	Not running	Yes	No
Standby	Running	Yes	No
Standby	Standby	Yes	No
Standby	Not Running	Yes	No
Not Running	Running	Yes	Yes
Not Running	Standby	Yes	No
Not Running	Not Running	Yes	Yes

Roadmap for Configuring Automatic Migration of Messaging/JMS-Related Services

WebLogic JMS leverages the migration framework by allowing an administrator to specify a migratable target for Messaging/JMS-related services, such as JMS servers and SAF agents. The WebLogic administrator can also configure migratable services so that will be automatically migrated from a failed server based on WebLogic Server health monitoring capabilities.

Note: JMS services can be migrated independent of the JTA Transaction Recovery Service. However, since the JTA Transaction Recovery Service provides the transaction control of the other subsystem services, it is usually migrated along with the other subsystem services. This ensures that the transaction integrity is maintained before and after the migration of the subsystem services.

To configure automatic JMS service migration on a migratable target within a cluster, perform the following tasks.

Step 1: Configured Managed Servers and Node Manager

Configure the Managed Servers in the cluster for migration, including assigning Managed Servers to a machine. In certain cases, Node Manager must also be running and configured to allow automatic server migration.

For step-by-step instructions for using the Administration Console to complete these tasks, refer to the following topics:

- Create Managed Servers

  Note: You must set a unique Listen Address value for the Managed Server instance that will host a migrated the JMS server; otherwise, the migration will fail.

- Create and configure machines
- Configure Node Manager

  Note: For automatic service migration, Consensus leasing requires that you use Node Manager to control servers within the cluster and that all migratable servers must have a Node Manager associated with them. For Database leasing, Node Manager is required only if pre-migration/post-migration scripts are defined. If pre/post-migrations are *not* defined, then Node manager is not required.

  For general information on configuring Node Manager, see Using Node Manager to Control Servers.

Step 2: Configure the Migration Leasing Basis

On the Cluster > Configuration > Migration page, configure the cluster's "Migration Basis" according to how your data persistence environment is configured, using either Database Leasing or Consensus Leasing. See "Leasing for Migratable Services" on page -12.

Step 3: Configure Migratable Targets

You should perform this step before targeting any Messaging/JMS-related services or enabling the JTA Transaction Recovery Service migration.

Configuring a Migratable Server As an Automatically Migratable Target

The Migratable Target Summary table in Administration Console displays the system-generated migratable targets of *servername* (migratable), which are automatically generated for each running server in a cluster. However, these are only generic templates and still need to be targeted and configured for automatic migration.

Create a New Migratable Target

When creating a new migratable target, the Administration Console provides a mechanism for creating, targeting, and selecting a migration policy.

Select a Preferred Server

When you create a new migratable target using the Administration Console, you can initially choose a preferred server in the cluster to associate the target with. The preferred server is the most appropriate server for hosting the migratable target.

Select an Auto Migration Policy

The default migration policy for all migratable targets is `manual`, so you need to select one of the following auto-migration policies:

- **exactly-once** – This policy indicates that if at least one Managed Server in the candidate list is running, then the service will be active somewhere in the cluster. It is important to note that this value can lead to target grouping. For example, if you have five `exactly-once` migratable targets and only boot one Managed Server in the cluster, then all five targets will be activated on that server.

- **failure-recovery** – This policy indicates that the service will only start if its User Preferred Server (UPS) is started. If the UPS is shutdown gracefully, this service will not be migrated anywhere. However, if the UPS crashes, is forcibly shutdown, or is otherwise made unavailable, then this service will be migrated to another candidate server.

See "Policies for Manual and Automatic Service Migration" on page -5.

Optionally Select Constrained Candidate Servers

When creating migratable targets that use the `exactly-once` services migration policy you may also want to restrict the potential member servers to which JMS servers can be migrated. A recommended best practice is to limit each migratable target's candidate server set to a primary, secondary, and perhaps tertiary server. Then as each server boots, the migratable targets will be restricted to their candidates rather than being satisfied by the first server to come online. Administrators can then manually migrate services to idle servers.

For the cluster's Path Service, however, the candidate servers for the migratable target should be the whole cluster, which is the default setting.

On the migratable target's Configuration > Migration page, the Constrained Candidate Servers **Available** box lists all the Managed Servers that could possibly support the migratable target. They become valid Candidate Servers when you move them into the **Chosen** box.

Optionally Specify Pre/Post-Migration Scripts

After creating a migratable target, you may also want to specify whether you are providing any pre-migration and post-migration scripts to perform any unmounting and mounting of the shared custom file store, as needed.

- **Pre-Migration Script Path** – the path to the pre-migration script to run before a migratable target is actually activated.
- **Post-Migration Script Path** – the path to the post-migration script to run after a migratable target is fully deactivated.
- **Post-Migration Script Failure Cancels Automatic Migration** – specifies whether or not a failure during execution of the post-deactivation script is fatal to the migration.
- **Allow Post-Migration Script To Run On a Different Machine** – specifies whether or not the post-deactivation script is allowed to run on a different machine.

The pre/post-migration scripts must be located in the `*BEA_HOME*/user_projects/domains/*mydomain*/bin/service_migration` directory, where `*mydomain*` is a domain-specific directory, with the same name as the domain. For your convenience, sample pre-migration and post-migrations scripts are provided in this directory.

Optionally Specify In-Place Restart Options

Migratable targets provide “Restart-In-Place” options to attempt to deactivate and reactivate a failed service, instead of migrating the service. See “In-Place Restarting of Failed Migratable Services” on page -14.

Step 4: Configure and Target Custom Stores

As discussed in “Custom Store Availability for JMS Services” on page -15, Messaging/JMS-related services require you to configure a custom persistent store that is also targeted to the same migratable targets as the JMS services, and make sure that the store is either:

- Configured such that all the candidate servers in a migratable target have access to the custom store
- Migrated around by pre/post migration scripts. See “Optionally Specify Pre/Post-Migration Scripts” on page -20.

Step 5: Target the JMS Services

When using migratable targets, you must target your JMS service to the same migratable target used by the custom persistent store. In the event that no custom store is specified for a JMS service that uses a migratable target, then a validation message will be generated, followed by failed JMS server deployment and a WebLogic Server boot failure. For example, attempting to target a JMS server that is using the default file store to a migratable target, will generate the following message:

```
Since the JMS server is targeted to a migratable target, it cannot use the
default store.
```

Similar messages are generated for a SAF agent or path service that is targeted to a migratable target and attempts to use the default store. In addition, if the custom store is not targeted to the same migratable target as the migratable service, then the following validation log message will be generated, followed by failed JMS server deployment and a WebLogic Server boot failure.

```
The JMS server is not targeted to the same target as its persistent store.
```

Special Considerations When Targeting SAF Agents or Path Service

There are some special targeting choices to consider when targeting SAF agents and a path service to migratable targets. For more information, see “Targeting Rules for SAF Agents” on page -9 and “Targeting Rules for Path Service” on page -10.

Step 6: Restart the Administration Server and Managed Servers With Modified Migration Policies

You must restart the Administration Server after configuring your JMS services for automatic service migration. You must also restart any Managed Servers whose migration policies were modified.

Step 7: Manually Migrating JMS Services Back to the Original Server

You may want to migrate a JMS service back to the original primary server once it is back online. Unlike the JTA Transaction Recovery Service, JMS services do not automatically migrate back to the primary server when it becomes available, so you need to manually migrate these services.

For instructions on manually migrating the Messaging/JMS-related services using the Administration Console, see Manually Migrate JMS Services in the *Administration Console Help*.

For instructions on manually migrating the Messaging/JMS-related services using WLST, see the WLST Command and Variable Reference in *WebLogic Scripting Tool*.

Roadmap for Configuring Manual Migration of Messaging/JMS-Related Services

WebLogic JMS leverages the migration framework by allowing an administrator to specify a migratable target for Messaging/JMS-related services. Once properly configured, a JMS service can be manually migrated to another WebLogic Server within a cluster. This includes both scheduled migrations as well as manual migrations in response to a WebLogic Server failure within the cluster.

To configure Messaging/JMS-related services for manual migration on a migratable target within a cluster, perform the following tasks.

Step 1: Configured Managed Servers

Configure the Managed Servers in the cluster for migration, including assigning Managed Servers to a machine.

For step-by-step instructions for using the Administration Console to complete these tasks, refer to the following topics:

- Create Managed Servers

  Note: You must set a unique Listen Address value for the Managed Server instance that will host a migrated the JMS server; otherwise, the migration will fail.

- Create and configure machines

Step 2: Configure Migratable Targets

You should perform this step before targeting any Messaging/JMS-related services or enabling the JTA Transaction Recovery Service migration.

Configuring a Migratable Server As a Migratable Target

The Migratable Target Summary table in Administration Console displays the system-generated migratable targets of *servername* (migratable), which are automatically generated for each running server in a cluster. However, these are only generic templates and still need to be targeted and configured for migration.

Create a New Migratable Target

When creating a new migratable target, the Administration Console provides a mechanism for creating, targeting, and selecting a migration policy.

Select a Preferred Server

When you create a new migratable target using the Administration Console, you can initially choose a preferred server in the cluster to associate the target with. The preferred server is the most appropriate server for hosting the migratable target.

Accept the Default Manual Policy

The default migration policy for all migratable targets is `manual`, so you do not need to change it.

Optionally Select Constrained Candidate Servers

When creating migratable targets you may also want to restrict the potential servers to which you can migrate Messaging/JMS-related services to only those that have access to a custom persistent store that is targeted to the same migratable target as the Messaging/JMS-related services.

For the cluster's Path Service, however, the candidate servers for the migratable target should be the whole cluster, which is the default setting.

On the migratable target's Configuration > Migration page, the Constrained Candidate Servers **Available** box lists all the Managed Servers that could possibly support the migratable target. They become valid Candidate Servers when you move them into the **Chosen** box.

Optionally Specify Pre/Post-Migration Scripts

After creating a migratable target, you may also want to specify whether you are providing any pre-migration and post-migration scripts to perform any unmounting and mounting of the shared custom store, as needed.

- **Pre-Migration Script Path** – the path to the pre-migration script to run before a migratable target is actually activated.
- **Post-Migration Script Path** – the path to the post-migration script to run after a migratable target is fully deactivated.
- **Post-Migration Script Failure Cancels Automatic Migration** – specifies whether or not a failure during execution of the post-deactivation script is fatal to the migration.
- **Allow Post-Migration Script To Run On a Different Machine** – specifies whether or not the post-deactivation script is allowed to run on a different machine.

The pre/post-migration scripts must be located in the `BEA_HOME/user_projects/domains/mydomain/bin/service_migration` directory, where `mydomain` is a domain-specific directory, with the same name as the domain. Basic directions for creating pre/post-migration scripts are provided in a `readme.txt` file in this directory.

Optionally Specify In-Place Restart Options

Migratable targets provide "Restart-In-Place" options to attempt to deactivate and reactivate a failed service, instead of migrating the service. See "In-Place Restarting of Failed Migratable Services" on page -14.

Step 3: Configure and Target Custom Stores

As discussed in "Custom Store Availability for JMS Services" on page -15, Messaging/JMS-related services require you to configure a custom persistent store that is also targeted to the same migratable targets as the JMS services, and make sure that the store is either:

- Configured such that all the candidate servers in a migratable target have access to the custom store
- Migrated around by pre/post migration scripts. See "Optionally Specify Pre/Post-Migration Scripts" on page -24.

Step 4: Target the JMS Services

When using migratable targets, you must target your JMS service to the same migratable target used by the custom persistent store. In the event that no custom store is specified for a JMS service that uses a migratable target, then a validation message will be generated, followed by failed JMS server deployment and a WebLogic Server boot failure. For example, attempting to target a JMS server that is using the default file store to a migratable target, will generate the following message:

```
Since the JMS server is targeted to a migratable target, it cannot use the
default store.
```

Similar messages are generated for a SAF agent or path service that is targeted to a migratable target and attempts to use the default store.

In addition, if the custom store is not targeted to the same migratable target as the migratable service, then the following validation log message will be generated, followed by failed JMS server deployment and a WebLogic Server boot failure.

```
The JMS server is not targeted to the same target as its persistent store.
```

Special Considerations When Targeting SAF Agents or Path Service

There are some special targeting choices to consider when targeting SAF agents and a path service to migratable targets. For more information, see "Targeting Rules for SAF Agents" on page -9 and "Targeting Rules for Path Service" on page -10.

Step 5: Restart the Administration Server and Managed Servers With Modified Migration Policies

You must restart the Administration Server after configuring your JMS services for manual service migration.

You must also restart any Managed Servers whose migration policies were modified.

Step 6: Manually Migrating JMS Services

For instructions on manually migrating the Messaging/JMS-related services using the Administration Console, see Manually Migrate JMS Services in the *Administration Console Help*.

For instructions on manually migrating the Messaging/JMS-related services using WLST, see the WLST Command and Variable Reference in *WebLogic Scripting Tool*.

Note: You may want to migrate a JMS service back to the original primary server once it is back online. Unlike the JTA Transaction Recovery Service, JMS services do not automatically migrate back to the primary server when it becomes available, so you need to manually migrate these services.

Roadmap for Configuring Automatic Migration of the JTA Transaction Recovery Service

The JTA Transaction Recovery Service is designed to gracefully handle transaction recovery after a crash. You can specify to have the Transaction Recovery Service automatically migrated from an unhealthy server instance to a healthy server instance, with the help of the server health monitoring services. This way the backup server can complete transaction work for the failed server.

To configure automatic migration of the Transaction Recovery Service for a migratable target within a cluster, perform the following tasks.

Step 1: Configured Managed Servers and Node Manager

Configure the Managed Servers in the cluster for migration, including assigning Managed Servers to a machine. Node Manager must also be running and configured to allow automatic server migration. The Node Manager is required to get liveliness information about the servers involved.

For step-by-step instructions for using the Administration Console to complete these tasks, refer to the following topics:

- Create Managed Servers

  Note: For information on configuring a primary server to not boot in Managed Server Independence (MSI) mode, which will prevent concurrent access to the TLOG with another backup server in recovery mode, see Managed Server Independence in *Programming WebLogic JTA*.

- Create and configure machines
- Configure Node Manager

  Note: For automatic service migration, Consensus leasing requires that you use Node Manager to control servers within the cluster and that all migratable servers must have a Node Manager associated with them. For Database leasing, Node Manager is required only if pre-migration/post-migration scripts are defined. If pre/post-migrations are *not* defined, then Node manager is not required.

  For general information on configuring Node Manager, see Using Node Manager to Control Servers.

Step 2: Configure the Migration Basis

On the Cluster > Configuration > Migration page, configure the cluster's "Migration Basis" according to how your data persistence environment is configured, using either Database Leasing or Consensus Leasing. See "Leasing for Migratable Services" on page -12.

Step 3: Enable Automatic JTA Migration

In the JTA Migration Configuration section on the Server > Configuration > Migration page, configure the following options.

Select the Automatic JTA Migration Check Box

Configure the automatic migration of the JTA Transaction Recovery Service by selecting the **Automatic JTA Migration Enabled** check box.

Optionally Select Candidate Servers

You may also want to restrict the potential servers to which you can migrate the Transaction Recovery Service to those that have access to the current server's transaction log files (stored in the default WebLogic store). If no candidate servers are chosen, then any server within the cluster can be chosen as a candidate server.

From the Candidate Servers **Available** box, select the Managed Servers that can access the JTA log files. They become valid Candidate Servers when you move them into the **Chosen** box.

Note: You must include the original server in the list of chosen servers so that you can manually migrate the Transaction Recovery Service back to the original server, if need be. The Administration Console enforces this rule.

Optionally Specify Pre/Post-Migration Scripts

You can specify whether you are providing any pre-migration and post-migration scripts to perform any unmounting and mounting of the shared storage, as needed.

- **Pre-Migration Script Path** – the path to the pre-migration script to run before a migratable target is actually activated.
- **Post-Migration Script Path** – the path to the post-migration script to run after a migratable target is fully deactivated.
- **Post-Migration Script Failure Cancels Automatic Migration** – specifies whether or not a failure during execution of the post-deactivation script is fatal to the migration.

- **Allow Post-Migration Script To Run On a Different Machine** – specifies whether or not the post-deactivation script is allowed to run on a different machine.

The pre/post-migration scripts must be located in the `BEA_HOME/user_projects/domains/mydomain/bin/service_migration` directory, where *`mydomain`* is a domain-specific directory, with the same name as the domain. Basic directions for creating pre/post-migration scripts are provided in a `readme.txt` file in this directory.

Step 4: Configure the Default Persistent Store For Transaction Recovery Service Migration

As discussed in "Default File Store Availability for JTA" on page -16, the Transaction Manager uses the default persistent store to store transaction log files. To enable migration of the Transaction Recovery Service, you must configure the default persistent store so that it stores its data files on a persistent storage solution that is available to other servers in the cluster if the original server fails.

Step 5: Restart the Administration Server and Managed Servers With Modified Migration Policies

You must restart the Administration Server after configuring the JTA Transaction Recovery service for automatic service migration.

You must also restart any Managed Servers whose migration policies were modified.

Step 6: Automatic Failback of the Transaction Recovery Service Back to the Original Server

After completing transaction recovery for a failed server, a backup server releases ownership of the Transaction Recovery Service so that the original server can reclaim it when the server is restarted. If the backup server stops (crashes) for any reason before it completes transaction recovery, its lease will expire. This way when primary server starts up, it can reclaim successfully ownership.

There are two scenarios for automatic failback of the Transaction Recovery Service to the primary server:

- Automatic failback *after* recovery is complete:

  - If the backup server finishes recovering the TLOG transactions before the primary server is restarted, it will initiate an implicit migration of the Transaction Recovery Service back to the primary server.
  - For both manual and automatic migration, the post-deactivation script would be executed automatically.
- Automatic failback *before* recovery is complete:
  - If the backup server is still recovering the TLOG transactions when the primary server is started, during the Transaction Recovery Service initialization of the primary server startup, it will initiate an implicit migration of the Transaction Recovery Service from the backup server.

Manual Migration of the JTA Transaction Recovery Service

The JTA Transaction Recovery Service is designed to gracefully handle transaction recovery after a crash. You can manually migrate the Transaction Recovery Service from an unhealthy server instance to a healthy server instance, with the help of the server health monitoring services. This way the backup server can complete transaction work for the failed server.

You can manually migrate the Transaction Recovery Service back to the original server by selecting the original server as the destination server. The backup server must not be running when you manually migrate the service back to the original server.

Note: Please note the following:

- If a backup server fails before completing the transaction recovery actions, the primary server cannot reclaim ownership of the Transaction Recovery Service and recovery will not be re-attempted on the rebooting server. Therefore, you must attempt to manually re-migrate the Transaction Recovery Service to another backup server.
- If you restart the original server while the backup server is recovering transactions, the backup server will gracefully release ownership of the Transaction Recovery Service. You do not need to stop the backup server. For detailed information, see Recovering Transactions For a Failed Clustered Server in *Programming WebLogic JTA*.
- For information on configuring a primary backup server to not boot in Managed Server Independence (MSI) mode, which will prevent concurrent access to the TLOG with another backup server in recovery mode, see Managed Server Independence in *Programming WebLogic JTA*.

For instructions on manually migrating the Transaction Recovery Service using the Administration Console, see "Migrate the Transaction Recovery Service" in the *Administration Console Help*.

Automatic Migration of User-Defined Singleton Services

Automatic singleton service migration allows the automatic health monitoring and migration of singleton services. A singleton service is a service operating within a cluster that is available on only one server at any given time. When a migratable service fails or become unavailable for any reason (for example, because of a bug in the service code, server failure, or network failure), it is deactivated at its current location and activated on a new server. The process of migrating these services to another server is handled via the Migration Master. See "Migration Master" on page -32.

WebLogic Server supports the automatic migration of user-defined singleton services.

Note: Although the JTA Transaction Recovery Service is also singleton service that is available on only one node of a cluster at any time, it is configured differently for automatic migration than user-defined singleton services. JMS and JTA services can also be manually migrated. See "Understanding the Service-Level Migration Framework" on page -3.

Overview of Singleton Service Migration

This section provides an overview of how automatic singleton service is implemented in WebLogic Server.

Migration Master

The migration master is a lightweight singleton service that monitors other services that can be migrated automatically. The server that currently hosts the migration master is responsible for starting and stopping the migration tasks associated with each migratable service.

Note: Migratable services do not have to be hosted on the same server as the migration master, but they must be hosted within the same cluster.

The migration master functions similar to the cluster master in that it is maintained by lease competition and runs on only one server at a time. Each server in a cluster continuously attempts to register the migration master lease. If the server currently hosting the migration master fails, the next server in the queue will take over the lease and begin hosting the migration master.

For more information on the cluster master, see "Cluster Master's Role in Server Migration" on page 7-18.

Note: The migration master and cluster master function independently and are not required to be hosted on the same server.

The server hosting the migration master maintains a record of all migrations performed, including the target name, source server, destination server, and the timestamp.

Migration Failure

If the migration of a singleton service fails on every candidate server within the cluster, the service is left deactivated. You can configure the number of times the number of times the migration master will iterate through the servers in the cluster.

Note: If you do not explicitly specify a list of candidate servers, the migration master will consider all of the cluster members as possible candidates for migration.

Implementing the Singleton Service Interface

Within an application, you can define a singleton service that can be used to perform tasks that you want to be executed on only one member of a cluster at any give time. The singleton service is implemented as a class within an application and is configured as part of the deployment descriptor on each server where the application is deployed. However, it is only active on one server at any time.

To create a singleton service within an application, you must create a class that, in addition to any tasks you wish the singleton service to perform, implements the `weblogic.cluster.singleton.SingletonService` interface.

The SingletonService interface contains the following methods which are used in the process of migration.

- public void activate()

  This method should obtain any system resources and start any services required for the singleton service to begin processing requests. This method is called in the following cases:

  – When a newly deployed application is started

  – During server start

  – During the activation stage of service migration

- public void deactivate()

  This method is called during server shutdown and during the deactivation stage of singleton service migration. This method should release any resources obtained through the activate() method. Additionally, it should stop any services that should only be available from one member of a cluster. The deactivate() method ensures that the singleton service lease is no longer held. The Migration Master will recognize that the lease is no longer held and will migrate the singleton service to another member of the cluster.

Note: Within an application, you can request immediate migration by calling the deactivate() method.

After you create a class that implements the SingletonService interface, you should ensure that this class is available in either the APP-INF/lib or APP-INF/classes directories when you create the `.war` file for deployment.

Deploying and Configuring Automatic Service Migration

After you create an application that implements the SingletonService interface, you must perform the following steps to deploy it and run it as a singleton service:

1. Deploy the application to a cluster.
2. Define a singleton service within WebLogic Server.
3. Configure the migration behavior of the singleton service.

The following sections outline these procedures in detail.

Deploying an Application as a Singleton Service

Although the singleton service will be active on only one cluster member at a time, you must deploy your application to every member of the cluster that will serve as a candidate target for the migrated singleton service.

After deploying the application to all of the candidate servers within the cluster, add the following entry is added to the weblogic-application.xml for each deployed instance of the application.

```
<weblogic-application>
...
   <singleton-service>
      <class-name>mypackage.MySingletonServiceImpl</class-name>
      <name>Appscoped_Singleton_Service</name>
   </singleton-service>
...
</weblogic-application>
```

Note: The `<class-name>` and `<name>` elements are required.

Defining a Singleton Service within WebLogic Server

After you have created and deployed your application, you must define a singleton service within WebLogic Server.

This singleton service object contains the following information:

- The path to the class to load as the singleton service
- The migration behavior of the singleton service

The singleton service object functions as a link between the migration master and the deployed application code. The following excerpt from the `cluster` element of `config.xml` shows how a singleton service is defined:

```
<SingletonService
   Name="SingletonTestServiceName"
   ClassName="mycompany.myprogram.subpackage.SingletonTestServiceImpl"
   Cluster="mycluster-"
/>
```

Configuring Automatic Service Migration

You can configure the behavior of automatic service migration using the following methods:

- WebLogic Server Administration Console

  The WebLogic Server Administration Console allows you to create and configure singleton services.

- WebLogic Scripting Tool (WLST)

  You can use the WebLogic Scripting Tool to configure automatic service migration using the MigratableTarget Management Bean.