Managing Serviceguard Extension for SAP Version B.05.00

Printed in the US
HP Part Number: T2803-90013
Published: March 2009

Legal Notices

Table of Contents

List of Figures

List of Tables

Printing History

Table 1 Editions and Releases

Printing Date	Part Number	Edition	SGeSAP Release	Operating System Releases
June 2000	B7885-90004	Edition 1	B.03.02	HP-UX 10.20 and HP-UX 11.00
March 2001	B7885-90009	Edition 2	B.03.03	HP-UX 10.20, HP-UX 11.00 and HP-UX 11i
June 2001	B7885-90011	Edition 3	B.03.04	HP-UX 10.20, HP-UX 11.00 and HP-UX 11i
March 2002	B7885-90013	Edition 4	B.03.06	HP-UX 11.00 and HP-UX 11i
June 2003	B7885-90018	Edition 5	B.03.08	HP-UX 11i
December 2004	T2357-90007	Edition 6	B.03.12	HP-UX 11i and HP-UX 11i v2
December 2005	T2357-90009	Edition 7	B.04.00	HP-UX 11i and HP-UX 11i v2
March 2006	T2803-90002	Edition 8	B.04.01	HP-UX 11i and HP-UX 11i v2
February 2007	T2803-90004	Edition 9	B.04.50	HP-UX 11i v3
November 2007	T2803-90011	Edition 10	B.04.02 / B.04.51	HP-UX 11i , HP-UX 11i v2 and HP-UX 11i v3
March 2009	T2803-90013	Edition 11	B.05.00	HP-UX 11i v2 and HP-UX 11i v3

The printing date and part number indicate the current edition. The printing date changes when a new edition is printed. (Minor corrections and updates which are incorporated at reprint do not cause the date to change.) The part number changes when extensive technical changes are incorporated.

New editions of this manual will incorporate all material updated since the previous edition.

HP Printing Division:

Business Critical Computing Business Unit
Hewlett-Packard Co.
19111 Pruneridge Ave.
Cupertino, CA 95014

About this Manual...

This document describes how to configure and install highly available SAP systems on HP-UX 11i v2 and HP-UX 11i v3 using Serviceguard. It refers to HP product T2803BA - Serviceguard Extension for SAP (SGeSAP).

To understand this document you have to have knowledge of the basic Serviceguard concepts and commands. Experience in the Basis Components of SAP will also be helpful.

This manual consists of six chapters:

- Chapter 1 "Understanding Serviceguard Extension for SAP" describes how to design a High Availability SAP Environment and points out how SAP components can be clustered.
- Chapter 2 "Planning the Storage Layout" proposes the recommended high available file system and shared storage layout for the SAP landscape and database systems.
- Chapter 3 "Step-by-Step Cluster Conversion" describes the installation of SGeSAP step-by-step down to the HP-UX command level.
- Chapter 4 "SAP Supply Chain Management" specifically deals with the SAP SCM and liveCache technology, gives a Storage Layout proposal and leads through the SGeSAP cluster conversion step-by-step.
- Chapter 5 "SAP Master Data Management" specifically deals with the SAP MDM technology and leads through the SGeSAP cluster conversion step-by-step.
- Chapter 6 "SGeSAP Cluster Administration" covers SGeSAP Administration aspects, as well as the use of different HP-UX platforms in a mixed cluster environment.

Table 2 Abbreviations

Abbreviation	Meaning
<SID>, <sid>, <XXSID>, <xxsid>	System ID of the SAP system, RDBMS or other components in uppercase/lowercase
<INSTNAME>	SAP instance, e.g. DVEBMGS, D, J, ASCS, SCS, ERS
[A]SCS	refers to either an SCS or an ASCS instance
<INSTNR>, <INR>	instance number of the SAP system
<primary>, <secondary>, <local>	names mapped to local IP addresses of the client LAN
<relocdb>, <relocci>, <relocdbci>	names mapped to relocatable IP addresses of SG packages in the client LAN
<primary_s>, <secondary_s>, <local_s>	names mapped to local IP addresses of the server LAN
<relocdb_s>, <relocci_s>, <relocdbci_s>	names mapped to relocatable IP addresses of Serviceguard packages in the server LAN
<...>	other abbreviations are self-explanatory and can be derived from the surrounding context

Related Documentation

The following documents contain additional related information:

- Serviceguard Extension for SAP Versions B.05.00 Release Notes (T2803-900012)
- Managing Serviceguard (B3936-90135)
- Serviceguard Release Notes (B3936-90119)
- Serviceguard NFS Toolkit A.11.11.08 and A.11.23.07 Release Notes (B5140-90032)
- Serviceguard NFS Toolkits A.11.31.03 Release Notes (B5140-90038)
- HP Storageworks RAID Manager XP user guide (T1610-96005)

1 Designing SGeSAP Cluster Scenarios

This chapter introduces the basic concepts used by the HP Serviceguard Extension for SAP (SGeSAP) and explains several naming conventions. The following sections provide recommendations and examples for typical cluster layouts that can be implemented for SAP environments:

- General Concepts of SGeSAP
- Mutual Failover Scenarios Using the Two Package Concept
- One Package Concept
- Follow-and-Push Clusters with Replicated Enqueue
- Dedicated NFS Packages
- Dialog Instance Clusters as Simple Tool for Adaptive Enterprises
- Handling of Redundant Dialog Instances
- Dedicated Failover Host

General Concepts of SGeSAP

SGeSAP extends HP Serviceguard's failover cluster capabilities to SAP application environments. SGeSAP continuously monitors the health of each SAP cluster node and automatically responds to failures or threshold violations. It provides a flexible framework of package templates to easily define cluster packages that protect various components of a mission-critical SAP infrastructure.

SGeSAP provides a flexible framework of package templates to easily define cluster packages that protect various components of a mission-critical SAP infrastructure. SGeSAP provides a single, uniform interface to cluster ABAP-only, JAVA-only, and add-in installations of SAP Web Application Servers (SAP WAS). Support includes SAP R/3 kernel, mySAP components, SAP Application Server for ABAP, SAP Application Server for JAVA, and SAP Netweaver based SAP applications in a range of supported release versions as specified in the separately available release notes.

The clustered SAP components include SAP ABAP Central Services, SAP JAVA Central Services, SAP ABAP Application Servers, SAP JAVA Application Servers, SAP Central Instances, SAP Enqueue Replication Servers, Oracle single-instance databases, MAXDB databases, SAP liveCache and SAP MDM components. For some platforms, support for liveCache hot standby clusters is included.

It is possible to combine all clustered components of a single SAP system into one failover package for simplicity and convenience. There is also full flexibility to split components up into several packages to avoid unwanted dependencies and to lower potential failover times.

Multiple SAP applications of different type and release version can be consolidated in a single cluster. SGeSAP enables SAP instance virtualization. It is possible to use SGeSAP to move redundant SAP ABAP Application Server Instances between hosts to quickly adapt to changing resource demands or maintenance needs. SGeSAP allows utilizing a combination of HP 9000 and HP Integrity servers in a mixed cluster with heterogeneous failover of SAP packages.

SAP applications can be divided into one or more distinct software components. Most of these components share a common technology layer, the SAP Application Server (SAPWAS). The SAP Application Server is the central building block of the SAP Netweaver technology. Each Application Server implementation comes with a characteristic set of software Single Points of Failure. These will become installed across the cluster hardware according to several high availability considerations and off-topic constraints, resulting in an individual configuration recommendation.

There are various publications available from SAP and third parties that describe the software components used by SAP applications in more detail. It is recommended to refer to these documents to get a basic familiarity before continuing to read. It is also recommended to familiarize with Serviceguard clustering and virtualization by reading the Serviceguard product manual "Managing Serviceguard", fifteenth edition or higher. The latest version can always be found at *http://docs.hp.com/en/ha.html#Serviceguard*.

Serviceguard packages can be distinguished into legacy packages and module-based packages. SGeSAP provides solutions for both approaches. SGeSAP consists of several SAP-related modules, legacy script

templates, SAP software service monitors as well as specialized additional features to integrate hot standby liveCache scenarios, HP Workload Management scenarios and HP Event Monitors.

There are three major Serviceguard modules delivered with SGeSAP. For the standard SAP Netweaver web application server stack it provides a Serviceguard module called `sgesap/sapinstance`. This module can be used to easily add a set of SAP instances that belong to the same Netweaver-based system to a module-based Serviceguard package. The package can encapsulate the failover entity for a combination of ABAP-stack, JAVA-stack and dual-stack instances plus, optionally, either Central Service Instances or Enqueue Replication Service Instances of an SAP System. For MAXDB or Oracle-based SAP database services, the module `sgesap/dbinstance` can be used. The module to cluster SAP liveCache instances is called `sgesap/livecache`. In addition to these three major modules, there are two more modules that enable easy clustering of smaller SAP infrastructure software tools `sgesap/sapinfra` and allow to manipulate the behavior of non-clustered SAP instances `sgesap/sapextinstance`. The covered infrastructure tools include the SAP `sapccmsr, saposcol, rfcadapter` and `saprouter` binaries. Other SGeSAP module names exist that provide a combination or subset of the functionality of some of the five modules mentioned above. They were primarily defined for convenience reasons to simplify configuration steps for standard use cases.

In legacy packaging each software Single Point of Failure defines a SGeSAP package type. SGeSAP follows a consistent naming convention for these package types. The naming conventions were created to be independent of SAP software release versions. This allows to use a similar approach for each SPOF, regardless of whether it appears in the latest SAP Netweaver stack or a SAP software that was released before the first design of SAP Application Server. Older SAP components sometimes only support a subset of the available clustering options.

Defining a mixture of legacy packages and module-based packages is possible in the same cluster. MDM packages, cross-subnet extensions for non-production use and SAP dispatcher monitoring are currently available in legacy packages only. Legacy-based packages will be discontinued at a later point in time. By then, all SGeSAP functionality will be available in a module version.

Table 1-1 Mapping the SGeSAP legacy package types to SGeSAP modules and different SAP naming conventions

SGeSAP legacy package type	SGeSAP module names	Commonly used SAP instance names
ci	`sgesap/sapinstance` `alternatives: sgesap/scs sgesap/ci`	DVEBMGS (as Central Instance), ASCS
jci		SCS
arep	`sgesap/sapinstance` `alternatives: sgesap/ers`	AREP, ENR, ERS
rep		REP, ENR, ERS
d	`sgesap/sapinstance`	D, DVEBMGS (new)
jd		JDI, JD, JC, J
db	`sgesap/dbinstance` `alternatives:sgesap/db sgesap/maxdb sgesap/oracledb`	
lc	`sgesap/livecache`	

Mutual Failover Scenarios Using the Two Package Concept

Most SAP applications rely on two central software services that define the major software Single Point of Failure (SPOF) for SAP environments: the SAP Enqueue Service and the SAP Message Service. These services are traditionally combined and run as part of a unique SAP Instance that is referred to as JAVA System Central Service Instance (SCS) for SAP JAVA applications or ABAP System Central Service Instance (ASCS) for SAP ABAP applications. If an SAP application has both JAVA and ABAP components, it is possible to have both - an SCS and an ASCS instance - for one SAP application. In this case, both instances are SPOFs that require clustering.

In pure ABAP environments, the term Central Instance (ci) is still in use for a software entity that combines further SAP application services with these SPOFs in a single instance. As any other SAP instance, a Central Instance has an Instance Name. Traditionally it is called DVEBMGS. Each letter represents a service that is delivered by the instance. The "E" and the "M" stand for the Enqueue and Message Service that were

identified as SPOFs in the system. Other SAP services can potentially be installed redundantly within additional Application Server instances, sometimes called Dialog Instances.

As its naming convention may suggest, DVEBMGS, there are more services available within the Central Instance than just those that cause the SPOFs. An undesirable result of this is, that a Central Instance is a complex software with a high resource demand. Shutdown and startup of Central Instances is slower and more error-prone than they could be. Starting with SAP Application Server 6.40 it is possible to isolate the SPOFs of the Central Instance in a separate Instance that is then called the ABAP System Central Service Instance, in short ASCS. The installer for SAP Application Server allows to install ASCS automatically. This installation procedure will then also create a standard Dialog Instance that is called DVEBMGS for compatibility reasons. This kind of DVEBMGS instance provides no Enqueue Service and no Message Service and is not a Central Instance anymore.

A package that uses the `sgesap/sapinstance` module can be set up to cluster the SCS and/or ASCS (or Central Instance) of a single SAP application.

The SGeSAP legacy ci package contains either a full DVEBMGS instance or a ASCS instance. The SGeSAP legacy `jci` packge contains a SCS instance. In any case, these packages provides failover capabilities to SAP Enqueue Services and SAP Message Services.

SAP application servers also require a database service, which usually defines the second software SPOF. SGeSAP bundles cluster capabilities for single-instance ORACLE RDBMS and SAP MAXDB RDBMS database services. The module `sgesap/dbinstance` (and similarly the legacy package type db) clusters any of these databases. The module unifies the configuration, so that database package administration for all database vendors is treated identically. `sgesap/dbinstance` can be used with any type of SAP application, independent of whether it is ABAP-based or JAVA-based or both. In case they are available, the module will take advantage of database tools that are shipped with certain SAP applications.

A SGeSAP legacy `jdb` package contains a database instance for SAP JAVA applications. A SGeSAP legacy `db` package contains a database instance for an ABAP application or a combined ABAP and JAVA application.

NOTE: It is not allowed to specify a single SGeSAP package with two database instances in it. An environment with `db` and `jdb` requires at least two packages to be defined.

If you are planning a simple three-tier SAP layout in a two node cluster, use the SGeSAP mutual failover model. This approach distinguishes two SGeSAP packages, one for the database SPOF and the other for the SAP SPOFs as defined above. In small and medium size environments, the database package gets combined with HA NFS server functionality to provide all filesystems that are required by the software in both packages. During normal operation, the two packages are running on different nodes of the cluster.

NOTE:

- Module-based SGeSAP database packages cannot be combined with a legacy based NFS toolkit to create a single package.
- The major advantage of this approach is, that the failed SAP package will never cause a costly failover of the underlying database since it is separated in a different package.
- It is not a requirement to do so, but it can help to reduce the complexity of a cluster setup, if SCS and ASCS are combined in a single package. Under these circumstances, it needs to be considered that the failure of one of the two instances will also cause failover for the other instance. This might be tolerable in those cases in which SAP replication instances are configured (see below).

The process of failover results in downtime that typically lasts a few minutes, depending on the work in progress when the failover takes place. A main portion of downtime is needed for the recovery of a database. The total recovery time of a failed database can not be predicted reliably.

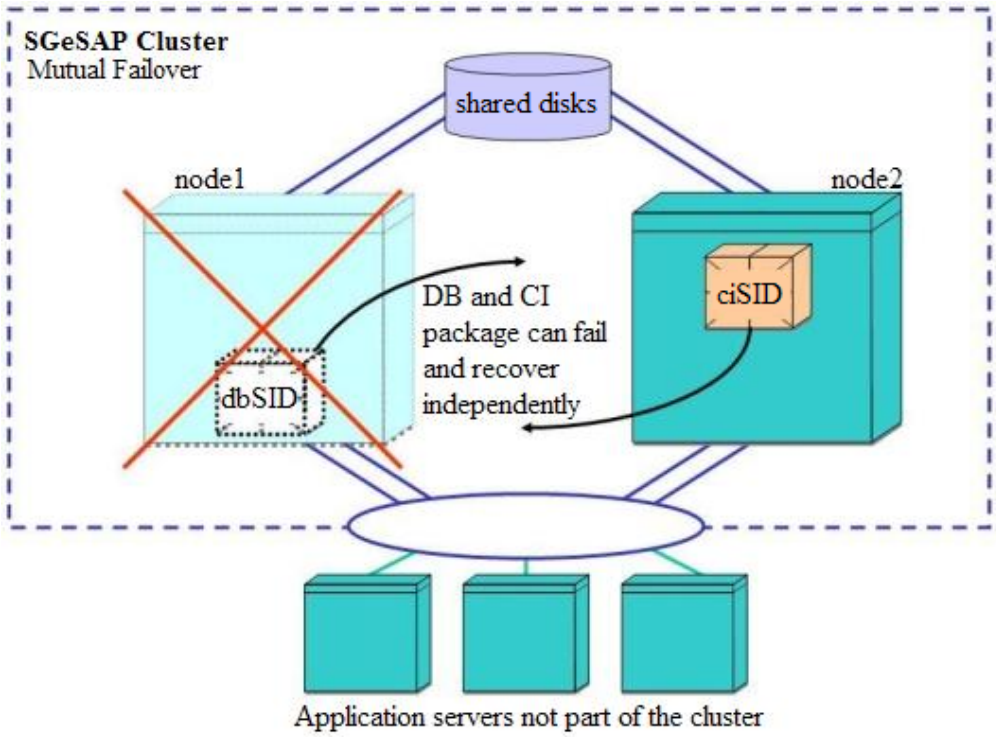
By tuning the Serviceguard heartbeat on a dedicated heartbeat LAN, it is possible to achieve failover times in the range of about a minute or two for a ci package that contains a lightweight [A]SCS instance without database.

NOTE: `sgesap/sapinstance` packages can identify the state of a corresponding `sgesap/dbinstance` package in the same cluster without the requirement of explicitly configuring Serviceguard package

dependencies. The information is for example used to delay SAP instance package startups while the database is starting in a separate package, but not yet ready to accept connections.

A cluster can be configured in a way that two nodes back up each other. The principle layout is depicted in figure 1-1. This picture as well as the following drawings are meant to illustrate basic principles in a clear and simple fashion. They omit other aspects and the level of detail that would be required for a reasonable and complete high availability configuration.

Figure 1-1 Two-Package Failover with Mutual Backup Scenario



It is a best practice to base the package naming on the SAP instance naming conventions whenever possible. Each package name should also include the SAP System Identifier (SID) of the system to which the package belongs. If similar packages of the same type get added later, they have a distinct namespace because they have a different SID.

Example: A simple mutual failover scenario for an ABAP application defines two packages, called dbSID and ascsSID (or ciSID for old SAP releases).

Robust Failover Using the One Package Concept

In a one-package configuration, the database, NFS and SAP SPOFs run on the same node at all times and are configured in one SGeSAP package. Other nodes in the Serviceguard cluster function as failover nodes for the primary node on which the system runs during normal operation.

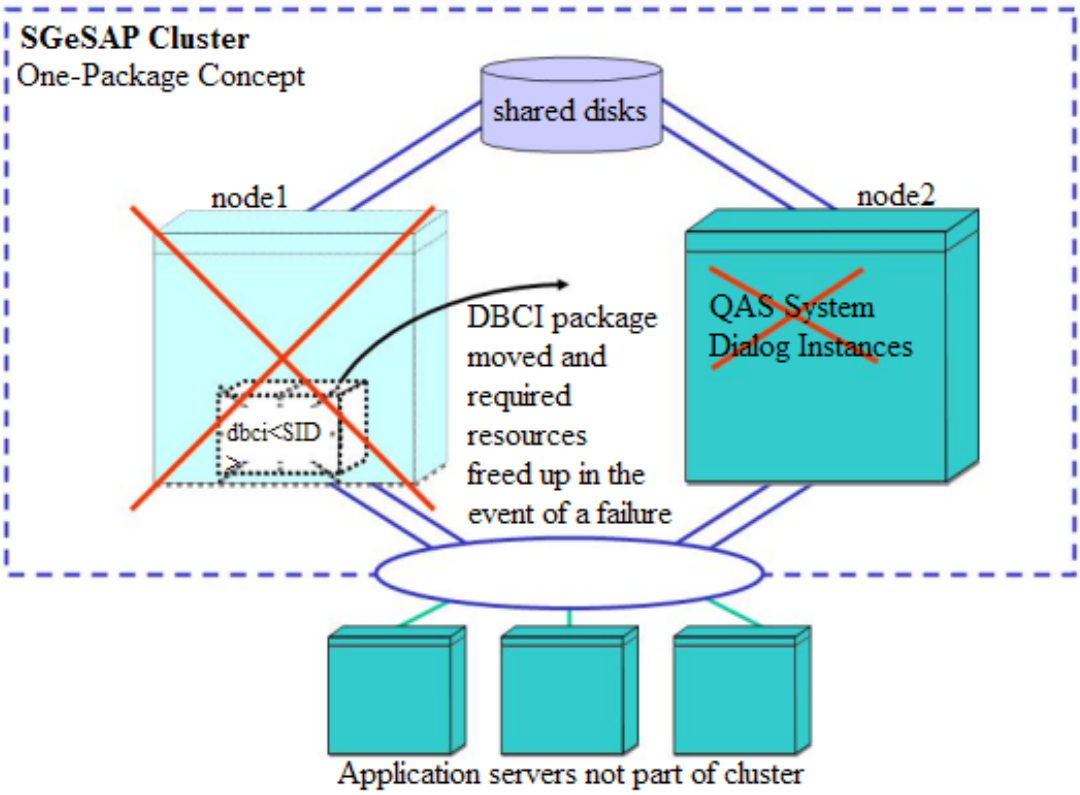
NOTE: Module-based SGeSAP packages cannot be combined with a legacy based NFS toolkit to create a single package.

It is not required to maintain an expensive idle standby. SGeSAP allows to utilize the secondary node(s) with different instances during normal operation. A common setup installs one or more non-mission critical SAP Systems on the failover nodes, typically SAP Consolidation, Quality Assurance or Development Systems. They can gracefully be shutdown by SGeSAP during failover to free up the computing resources for the critical production system. For modular packages, the `sgesap/sapextinstance` module can be added to the package to allow specifying this kind of behavior.

Development environments tend to be less stable than production systems. This should be taken into consideration before mixing these use-cases in a single cluster. A feasible alternative would be to install Dialog Instances of the production system on the failover node.

If the primary node fails, the database and the Central Instance fail over and continue functioning on an adoptive node. After failover, the system runs without any manual intervention needed. All redundant Application Servers and Dialog Instances, even those that are not part of the cluster, can either stay up or be restarted triggered by a failover. A sample configuration in Figure 1-2 shows node1 with a failure, which causes the package containing the database and central instance to fail over to node2. A Quality Assurance System and additional Dialog Instances get shut down, before the database and Central Instance are restarted.

Figure 1-2 One-Package Failover Scenario



Follow-and-Push Clusters with Replicated Enqueue

In case an environment has very high demands regarding guaranteed uptime, it makes sense to activate a Replicated Enqueue with SGeSAP. With this additional mechanism it is possible to failover ABAP and/or JAVA System Central Service Instances without impacting ongoing transactions on Dialog Instances.

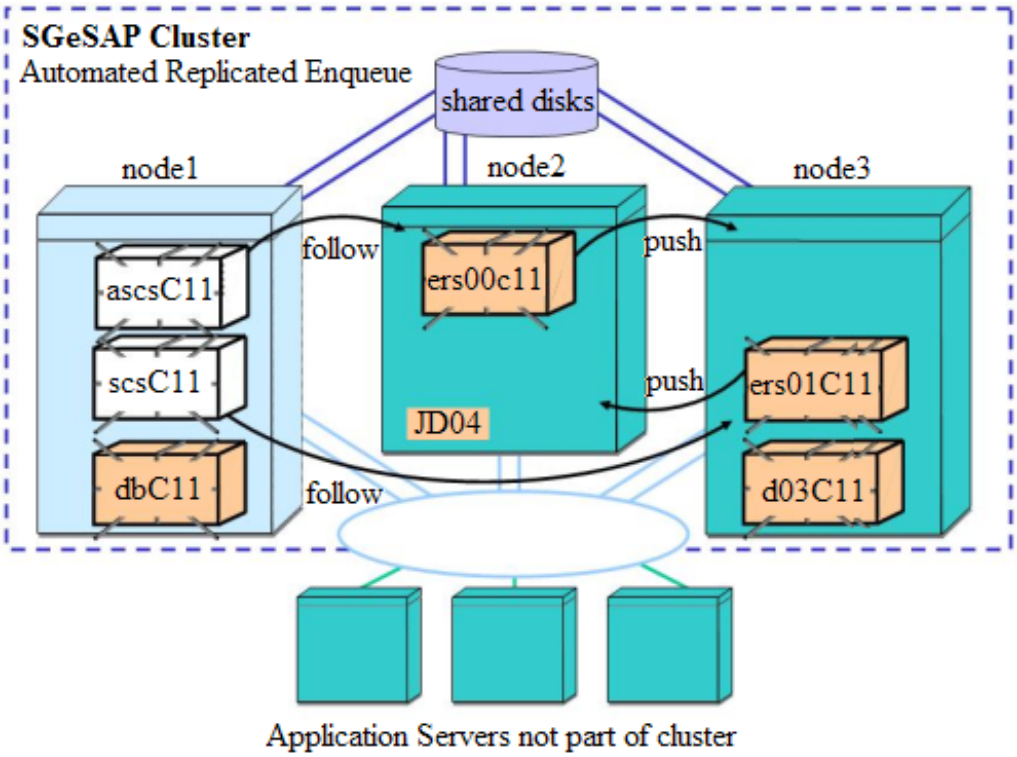
NOTE: It only makes sense to activate Enqueue Replication for systems that have Dialog Instances that run on nodes different from the primary node of the System Central Service package.

Each SAP Enqueue Service maintains a table of exclusive locks that can temporarily be granted exclusively to an ongoing transaction. This mechanism avoids inconsistencies that could be caused by parallel transactions that access the same data in the database simultaneously. In case of a failure of the Enqueue Service, the table with all locks that have been granted gets lost. After package failover and restart of the Enqueue Service, all Dialog Instances need to get notified that the lock table content got lost. As a reaction they cancel ongoing transactions that still hold granted locks. These transactions need to be restarted.

Enqueue Replication provides a concept that prevents the impact of a failure of the Enqueue Service on the Dialog Instances. Transactions no longer need to be restarted. The Enqueue Server has the ability to create a copy of its memory content to a Replicated Enqueue Service that needs to be running as part of a Enqueue Replication Service Instance (ERS) on a remote host. This is a real-time copy mechanism and ensures that the replicated memory accurately reflects the status of the Enqueue Server at all times.

There might be two ERS instances for a single SAP system, replicating SCS and ASCS locks separately.

Figure 1-3 Replicated Enqueue Clustering for ABAP and JAVA Instances



Enqueue Services also come as integral part of each ABAP DVEBMGS Central Instance. This integrated version of the Enqueue Service is not able to utilize replication features. The DVEBMGS Instance needs to be split up in a standard Dialog Instance and a ABAP System Central Service Instance (ASCS).

The SGeSAP packaging of the ERS Instance provides startup and shutdown routines, failure detection, split-brain prevention and quorum services to the mechanism. SGeSAP also delivers an EMS (HP-UX Event Monitoring Service) that implements a cluster resource called

`/applications/sap/enqor/<SID >ers<INSTNR>`

for each Replicated Enqueue in the cluster. Monitoring requests can be created to regularly poll the status of each Replicated Enqueue.

NOTE: For SAP versions that were released before the ERS naming conventions got introduced, the resource is also offered, but called `/applications/sap/enqor/<SID>[a]scs`.

The EMS monitor can be used to define a resource in the Serviceguard packages. This implements a follow-and-push behavior for the two packages that include enqueue and its replication. As a result, an automatism will make sure that enqueue and its replication server are never started on the same node initially. Enqueue will not invalidate the replication accidentally by starting on a non-replication node while replication is active elsewhere. It is possible to move the package with the replication server to any free node in a multi-node cluster without a requirement to reconfigure the enqueue package failover policy.

During failover of enqueue, its replication will be located dynamically and the enqueue restarts on the currently active replication node. Enqueue synchronizes with the local replication server. As a next step, the package with the replication service shuts down automatically and restarts on a healthy node, if available. In case of a failover in a multi-node environment this implements a self-healing capability for the replication function. Enqueue will failover to just any node from the list of statically configured hosts if no replication package is running.

Two replication instances are required if Enqueue Replication Services are to be used for both the JAVA stack and the ABAP stack. From this approach several configuration options derive. In most cases, it is the best practice to create separate packages for ASCS, SCS and the two ERS instances. It is also supported to combine the replication instances within one SGeSAP package. It is also supported to combine ASCS and SCS in one package, but only if the two ERS instances are likewise combined in another package. It is not supported to combine ASCS and SCS in one package and keep the two ERS instances in two separate

packages. Otherwise, situations can arise in which a failover of the combined ASCS/SCS package is not possible. Finally, ASCS can not be combined with its ERS instance (AREP) in the same package. For the same reason, SCS can not be combined with its ERS instance (REP).

The `sgesap/sapinstance` module can be used to cluster Enqueue Replication Instances. Furthermore, SGeSAP offers the legacy package types rep and arep to implement enqueue replication packages for JAVA and ABAP.

SAP offers two possibilities to configure Enqueue Replication Servers:

1. SAP self-controlled using High Availability polling
2. Completely High Availability failover solution controlled

SGeSAP provides a completely High Availability failover solution controlled implementation that avoids costly polling data exchange between SAP and the High Availability cluster software. There are several SAP profile parameters that are related to the self-controlled approach. Most of these parameters have names that start with the string `enque/enrep/hafunc_`. They will not have any effect in SGeSAP clusters.

Dedicated NFS Packages

Small clusters with only a few SGeSAP packages usually provide HA NFS by combining the HA NFS toolkit package functionality with the SGeSAP packages that contain a database component. The HA NFS toolkit is a separate product with a set of configuration and control files that must be customized for the SGeSAP environment. It needs to be obtained separately.

HA NFS is delivered in a distributed fashion with each database package serving its own filesystems. By consolidating this into one package, all NFS serving capabilities can be removed from the database packages.In complex, consolidated environments with several SGeSAP packages it is of significant help to use one dedicates HA NFS package instead of blending this into existing packages.

A dedicated `SAPNFS` package is specialized to provide access to shared filesystems that are needed by more than one mySAP component. Typical filesystems served by `SAPNFS` would be the common SAP transport directory or the global `SAPDB` executable directory. The `SAPDB` client libraries are part of the global `SAPDB` executable directory and access to these files is needed by APO and liveCache at the same time.

SGeSAP setups are designed to avoid HA NFS shared filesystems with heavy traffic if possible. For many implementations, this gives the option to use one `SAPNFS` package for all HA NFS needs in the SAP consolidation cluster without the risk to create a serious performance bottleneck.

HA NFS might still be required in configurations that use Cluster File Systems in order to provide access to the SAP transport directories to SAP instances that run on hosts outside of the cluster.

Dialog Instance Clusters as Simple Tool for Adaptive Enterprises

Databases and Central Instances are Single Points of Failure. ABAP and JAVA Dialog Instances can be installed in a redundant fashion. In theory, this allows to avoid additional SPOFs in Dialog Instances. This doesn't mean that it is impossible to configure the systems including SPOFs on Dialog Instances. A simple example for the need of a SAP Application Server package is to protect dedicated batch servers against hardware failures.

Any number of SAP Application Server instances can be added to a package that uses the module `sgesap/sapinstance`. SAP ABAP Dialog Instances can also be packages in SGeSAP legacy package type 'd'. SAP JAVA Dialog Instances can be packaged using SGeSAP legacy package type 'jd'.

Dialog Instance packages allow an uncomplicated approach to achieve abstraction from the hardware layer. It is possible to shift around Dialog Instance packages between servers at any given time. This might be desirable if the CPU resource consumption is eventually balanced poorly due to changed usage patterns. Dialog Instances can then be moved between the different hosts to address this. A Dialog Instance can also be moved to a standby host to allow planned hardware maintenance for the node it was running on.

One can simulate this flexibility by installing Dialog Instances on every host and activating them if required. This might be a feasible approach for many purposes and saves the need to maintain virtual IP addresses for each Dialog Instance. But there are ways that SAP users unintentionally create additional short-term SPOFs during operation if they reference a specific instance via its hostname. This could e.g. be done during batch scheduling. With Dialog Instance packages, the system becomes invulnerable against this type of user error.

Dialog Instance virtualization packages provide high availability and flexibility at the same time. The system becomes more robust using Dialog Instance packages. The virtualization allows moving the instances manually between the cluster hosts on demand.

Figure 1-4 Failover Node with Application Server package

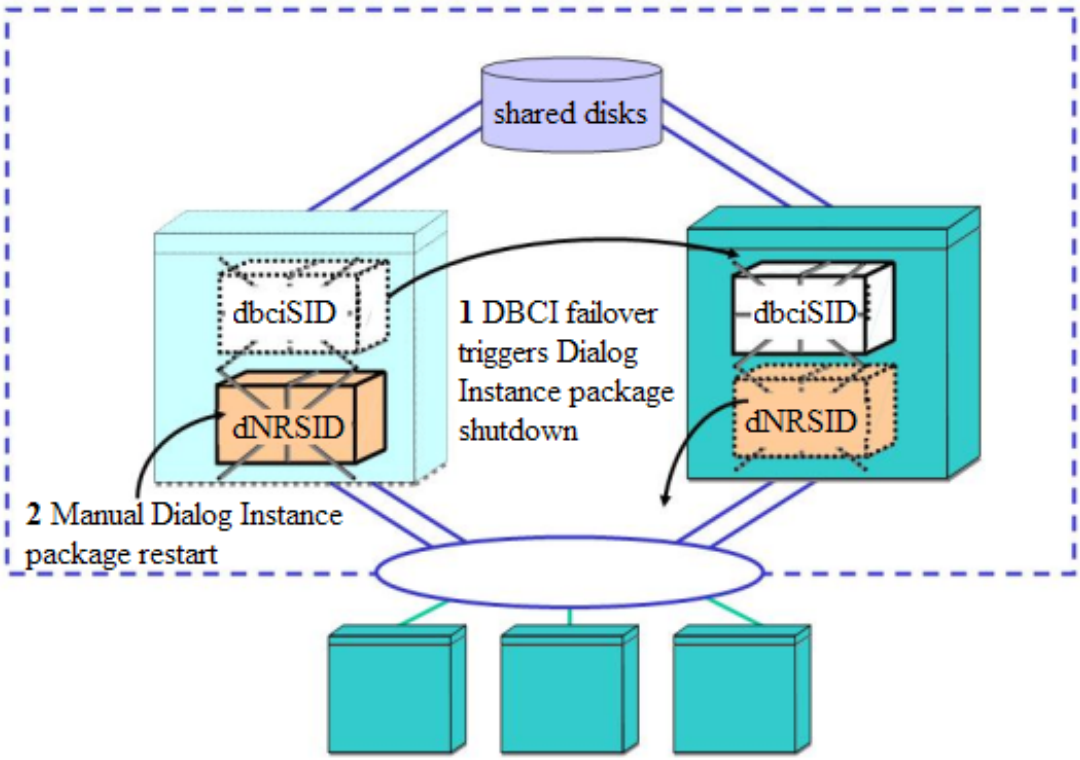


Figure 1-4 illustrates a common configuration with the adoptive node running as a Dialog Server during normal operation. Node1 and node2 have equal computing power and the load is evenly distributed between the combination of database and Central Service Instance on node1 and the additional Dialog Instance on node2. If node1 fails, the Dialog Instance package will be shut down during failover of the `dbciSID` package. This is similar to a one-package setup without Dialog Instance packaging.

The advantage of this setup is, that after repair of node1, the Dialog Instance package can just be restarted on node1 instead of node2. This saves downtime that would otherwise be necessary caused by a failback of the `dbciSID` package. The two instances can be separated to different machines without impacting the production environment negatively. It should be noted that for this scenario with just two hosts there is not necessarily a requirement to enable automatic failover for the Dialog Instance package.

The described shutdown operation for Dialog Instance packages can be specified in any SGeSAP legacy package directly. In modularized SGeSAP it is recommended to use generic Serviceguard package dependencies instead.

Handling of Redundant Dialog Instances

Non-critical SAP Application Servers can be run on HP-UX, SUSE or RedHat LINUX application server hosts. These hosts do not need to be part of the Serviceguard cluster. Even if the additional SAP services are run on nodes in the Serviceguard cluster, they are not necessarily protected by Serviceguard packages. A combination of Windows/HP-UX application servers is technically possible but additional software is required to access HP-UX filesystems or HP-UX-like remote shells from the Windows system.

All non-packaged ABAP instances are subsequently called Additional Dialog Instances or sometimes synonymously Additional SAP Application Servers to distinguish them from mission-critical Dialog Instances. An additional Dialog Instance that runs on a cluster node is called an Internal Dialog Instance. External Dialog Instances run on HP-UX or Linux hosts that are not part of the cluster. Even if Dialog Instances are external to the cluster, they may be affected by package startup and shutdown.

For convenience, Additional Dialog Instances can be started, stopped or restarted with any SGeSAP package that secures critical components. Some SAP applications require the whole set of Dialog Instances to be restarted during failover of the Central Service package. This can be triggered with SGeSAP means.

It helps to better understand the concept, if one considers that all of these operations for non-clustered instances are considered inherently non-critical. If they fail, this failure won't have any impact on the ongoing package operation. A best-effort attempt is made, but there are no guarantees that the operation succeeds. If such operations need to succeed, package dependencies in combination with SGeSAP Dialog Instance packages need to be used.

Dialog Instances can be marked to be of minor importance. They will then be shut down, if a critical component fails over to the host they run on in order to free up resources for the non-redundant packaged components. Additional Dialog Instances never get reflected in package names.

The described functionality can be achieved by adding the module `sgesap/sapextinstance` to the package. Legacy SGeSAP provides similar functionality, but SAP JAVA instances are not handled.

NOTE: Declaring non-critical Dialog Instances in a package configuration doesn't add them to the components that are secured by the package. The package won't react to any error conditions of these additional instances. The concept is distinct from the Dialog Instance packages that got explained in the previous section.

If Additional Dialog Instances are used, certain rules should be followed:

Use `saplogon` with Application Server logon groups. When logging on to an application server group with two or more Dialog Instances, you don't need a different login procedure even if one of the Application Servers of the group fails. Also, using login groups provides workload balancing between Application Servers.

Avoid specifying a destination host when defining a batch job. This allows the batch scheduler to choose a batch server that is available at the start time of the batch job. If you must specify a destination host, specify the batch server running on the Central Instance or a packaged Application Server Instance.

Print requests stay in the system until a node is available again and the Spool Server has been restarted. These requests could be moved manually to other spool servers if one spool server is unavailable for a long period of time. An alternative is to print all time critical documents through the highly available spool server of the central instance.

Configuring the Update Service as part of the packaged Central Instance is recommended. Consider using local update servers only if performance issues require it. In this case, configure Update Services for application services running on the same node. This ensures that the remaining SAP Instances on different nodes are not affected if an outage occurs on the Update Server. Otherwise a failure of the Update Service will lead to subsequent outages at different Dialog Instance nodes.

Dedicated Failover Host

More complicated clusters that consolidate a couple of SAP applications often have a dedicated failover server. While each SAP application has its own set of primary nodes, there is no need to also provide a failover node for each of these servers. Instead there is one commonly shared secondary node that is in principle capable to replace any single failed primary node. Often, some or all of the primary nodes are partitions of a large server.

Figure 1-5 Replicated Enqueue Clustering for ABAP and JAVA Instances

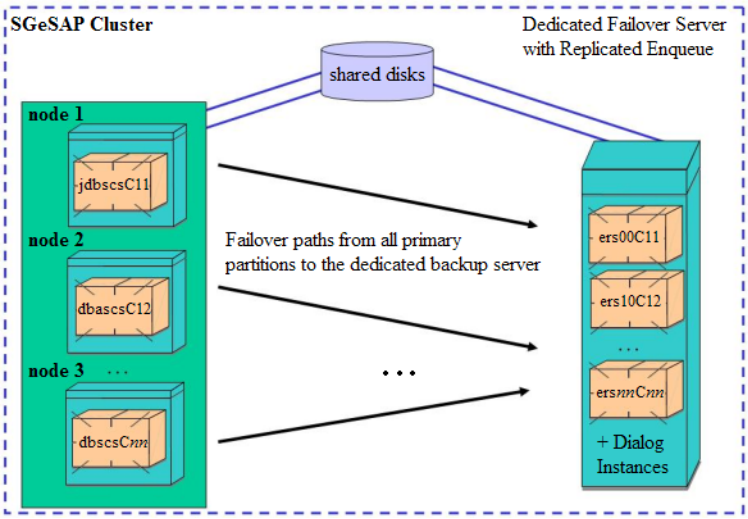


Figure 1-5 shows an example configuration. The dedicated failover host can serve many purposes during normal operation. With the introduction of Replicated Enqueue Servers, it is a good practice to consolidate a number of Replicated Enqueues on the dedicated failover host.

These replication units can be halted at any time without disrupting ongoing transactions for the systems they belong to. They are ideally sacrificed in emergency conditions in which a failing database and/or Central Service Instance need the spare resources.

2 Planning the Storage Layout

Volume managers are tools that let you create units of disk storage known as storage groups. Storage groups contain logical volumes for use on single systems and in high availability clusters. In Serviceguard clusters, package control scripts activate storage groups. Two volume managers can be used with Serviceguard: the standard Logical Volume Manager (LVM) of HP-UX and the Veritas Volume Manager (VxVM). SGeSAP can be used with both volume managers. The following steps describe two standard setups for the LVM volume manager. VxVM setups can be configured accordingly. A third storage layout option describes a Cluster File System configuration for SGeSAP.In this case, VxVM must be used and all Application Servers need to run on cluster nodes. Chapter three explores the concepts and details the implementation steps discussed in this chapter.

Database storage layouts for usage with parallel databases are only briefly described for Oracle Real Application Clusters. Detailed configuration steps for parallel database technologies are not covered in this manual. Additional information about SGeSAP and parallel databases is being released as whitepapers from HP. Refer to the Additional Reading section of the relevant SGeSAP release notes to verify the availability of whitepapers in this area.

This chapter discusses disk layout for clustered SAP components and database components of several vendors on a conception level. It is divided into two main sections:

- SAP Instance Storage Considerations
- Database Instance Storage Considerations

SAP Instance Storage Considerations

In general, it is important to stay as close as possible to the original layout intended by SAP. But certain cluster specific considerations might suggest a slightly different approach in some cases. SGeSAP supports various combinations of providing shared access to file systems in the cluster. The possible storage layout and file system configuration options include:

Each filesystem that gets added by the SAP installation routines needs to be classified and a decision has to be made.

Refer to table 2-1 for more information.

Table 2-1 Option descriptions

Option:	Description
1 - SGeSAP NFS Cluster	optimized to provide maximum flexibility. Following the recommendations given below allows for expansion of existing clusters without limitations caused by the cluster. Another important design goal of SGeSAP option 1 is, that a redesign of the storage layout is not imperative when adding additional SAP components later on. Effective change management is an important aspect for production environments. The disk layout needs to be as flexible as possible to allow growth to be done by just adding storage for newly added components. If the design is planned carefully at the beginning, it is not required to make changes to already existing file systems. Option 1 is recommended for environments that implement clusters with server consolidation if CFS is not available.
2 - SGeSAP NFS Idle Standby Cluster	optimized to provide maximum simplicity. The option is only feasible for very simple clusters. It needs to be foreseeable that their layout and configuration won't change over time. It comes with the disadvantage of being locked into restricted configurations with a single SAP System and idle standby nodes. HP recommends option 1 in case of uncertainty about potential future layout changes.
3 - SGeSAP CFS Cluster	combines maximum flexibility with the convenience of a Cluster File System. It is the most advanced option. CFS should be used with SAP if available. The HP Serviceguard Cluster File System requires a set of multi-node packages. The number of packages varies with the number of disk groups and mountpoints for Cluster File Systems. This can be a limiting factor for highly consolidated SGeSAP environments.

Each filesystem that gets added to a system by SAP installation routines needs to be classified and a decision has to be made:

- Whether it needs to be kept as a local copy on internal disks of each node of the cluster.
- Whether it needs to be shared on a SAN storage device to allow failover and exclusive activation.
- Whether it needs to provide shared access to more than one node of the cluster at the same time.

NOTE: SGeSAP packages and service monitors require SAP tools. Patching the SAP kernel sometimes also patches SAP tools. Depending on what SAP changed, this might introduce additional dependencies on shared libraries that weren't required before the patch. Depending on the SHLIB_PATH settings of the root user it might no longer be possible for SGeSAP to execute the SAP tools after applying the patch. The introduced additional libraries are not found. Creating local copies of the complete central executable directory prevents this issue.

The following sections detail the three different storage options.

Option 1: SGeSAP NFS Cluster

With this storage setup SGeSAP makes extensive use of exclusive volume group activation. Concurrent shared access is provided via NFS services. Automounter and cross-mounting concepts are used in order to allow each node of the cluster to switch roles between serving and using NFS shares. It is possible to access the NFS file systems from servers outside of the cluster, which is an intrinsic part of many SAP configurations.

Common Directories that are Kept Local

The following common directories and their files are kept local on each node of the cluster:

- `/home/<SID>adm` — the home directory of the SAP system administrator with node specific startup log files
- `/usr/sap/<SID>/SYS/exe/run` — the directory that holds a local copy of all SAP instance executables, libraries and tools (optional for kernel 7.x and higher)
- `/usr/sap/tmp` — the directory in which the SAP operating system collector keeps monitoring data of the local operating system
- `/usr/sap/hostctrl` — the directory in which SAP control services for the local host are kept (kernel 7.x and higher)
- `/etc/cmcluster` — the directory in which Serviceguard keeps its legacy configuration files and the node specific package runtime directories
- Depending on database vendor and version, it might in addition be required to keep local database client software. Details can be found in the database sections below.

Part of the content of the local group of directories must be synchronized manually between all nodes of the cluster. Serviceguard provides a tool `cmcp(1m)` that allows easy replication of a file to all cluster nodes.

SAP instance (startup) profile names contain either local hostnames or virtual hostnames. SGeSAP will always prefer profiles that use local hostnames to allow individual startup profiles for each host, which might be useful if the failover hardware differs in size.

In clustered SAP environments prior to 7.x releases it is required to install local executables. Local executables help to prevent several causes for package startup or package shutdown hangs due to the unavailability of the centralized executable directory. Availability of executables delivered with packaged SAP components is mandatory for proper package operation. Experience has shown that it is a good practice to create local copies for all files in the central executable directory. This includes shared libraries delivered by SAP.

NOTE: SGeSAP packages and service monitors require SAP tools. Patching the SAP kernel sometimes also patches SAP tools. Depending on what SAP changed, this might introduce additional dependencies on shared libraries that weren't required before the patch. Depending on the `SHLIB_PATH` settings of the root user it might no longer be possible for SGeSAP to execute the SAP tools after applying the patch. The introduced additional libraries are not found. Creating local copies of the complete central executable directory prevents this issue.

To automatically synchronize local copies of the executables, SAP components deliver the `sapcpe` mechanism. With every startup of the instance, `sapcpe` matches new executables stored centrally with those stored locally.

Directories that Reside on Shared Disks

Volume groups on SAN shard storage are configured as part of the SGeSAP packages. They can be either:

- instance specific or
- system specific or
- environment specific.

Instance specific volume groups are required by only one SAP instance or one database instance. They usually get included with exactly the package that is set up for this instance.

System specific volume groups get accessed from all instances that belong to a particular SAP System. Environment specific volume groups get accessed from all instances that belong to all SAP Systems installed in the whole SAP environment. System and environment specific volume groups are set up using HA NFS to provide access for all instances. They shouldn't be part of a package that is only dedicated to a single SAP instance if there are several of them. If this package is down, then other instances would also be impacted. As a rule of thumb, it is a good default to put all these volume groups into a package that holds the database of the system. These filesystems often provide tools for database handling that don't require the SAP instance at all.

In consolidated environments with more than one SAP application component it is recommended to separate the environment specific volume groups to a dedicated HA NFS package. This package will be referred to as sapnfs package. It should remain running all the time, since it is of central importance for the whole setup. Since `sapnfs` is just serving networked file systems, there is rarely needed to stop this package at any time. If environment specific volume groups become part of a database package, there will be a potential dependency between packages of different SAP Systems. Stopping one SAP System by halting all related Serviceguard packages will then lead to a lack of necessary NFS resources for otherwise unrelated SAP Systems. The `sapnfs` package avoids this unpleasant dependency. It is an option to also move the system specific volume groups to the `sapnfs` package. This can be done, to keep HA NFS mechanisms completely separate.

A valuable naming convention for most of these shared volume groups is `vg<INSTNAME><SID>` or alternatively `vg<INSTNAME><SID><INR>`. Table 2-2 and Table 2-3 provide an overview of SAP shared storage and maps them to the component and package type for which they occur.

Table 2-2 Instance Specific Volume Groups for exclusive activation with a package

Mount Point	Access Point	Recommended packages	VG Name	Device minor number
/usr/sap/<SID>/SCS<INR>	Shared disk	jci<SID> (scs<SID>) jdbjci<SID>		
/usr/sap/<SID>/ASCS<INR>		ci<SID> (ascs<SID>) dbci<SID>		
/usr/sap/<SID>/ERS<INR>		ers<INR><SID>		
/usr/sap/<SID>/DVEBMGS<INR>		ci<SID> dbci<SID> d<INR><SID> (SAP kernel 7.x)		
/usr/sap/<SID>/D<INR>		d<INR><SID>		

Table 2-3 System and Environment Specific Volume Groups

Mount Point	Access Point	Potential owning packages	VG Name	Device minor number
/export/sapmnt/<SID>	shared disk and HA NFS	db<SID> dbci<SID> jdb<SID> jdbjci<SID> sapnfs		
/export/usr/sap/trans		db<SID> dbci<SID> sapnfs		
/usr/sap/put	shared disk	none		

The tables can be used to document used device minor numbers. The device minor numbers of logical volumes need to be identical for each distributed volume group across all cluster nodes.

/usr/sap/<SID> should not be added to a package, since using this as a dynamic mount point would prohibit access to the instance directories of locally installed additional SAP application servers. The /usr/sap/<SID> mount point will also be used to store local SAP executables. This prevents problems with busy mount points during database package shutdown. Due to the size of the directory content, it should not be part of the local root file system. The /usr/sap/tmp might or might not be part of the root file system. This is the working directory of the operating system collector process saposcol. The size of this directory will rarely be beyond a few Megabytes.

If you have more than one system, place /usr/sap/put on separate volume groups created on shared drives. The directory should not be added to any package. This ensures that they are independent from any SAP WAS system and you can mount them on any host by hand if needed.

All filesystems mounted below /export are part of HA NFS cross-mounting via automounter. The automounter uses virtual IP addresses to access the HA NFS directories via the path that comes without the /export prefix. This ensures that the directories are quickly available after a switchover. The cross-mounting allows coexistence of NFS server and NFS client processes on nodes within the cluster.

Option 2: SGeSAP NFS Idle Standby Cluster

This option has a simple setup, but it is severely limited in flexibility. In most cases, option 1 should be preferred. A cluster can be configured using option 2 if it fulfills all of the following prerequisites:

- Only one SGeSAP package is configured in the cluster. Underlying database technology is a single-instance Oracle RDBMS. The package combines failover services for the database and all required NFS services and SAP central components (ABAP CI, SCS, ASCS). There are no Application Server Instances installed on cluster nodes. Replicated Enqueue is not in use.
- There is no additional SAP software installed on the cluster nodes

The use of a HA NFS service can be configured to export file systems to external Application Servers that manually mount them. A dedicated NFS package is not possible. Dedicated NFS requires option 1.

Common Directories that are Kept Local

The following common directories and their files are kept local on each node of the cluster:

- /home/<SID>adm — the home directory of the SAP system administrator with node specific startup log files
- /usr/sap/<SID>/SYS/exe/run — the directory that holds a local copy of all SAP instance executables, libraries and tools (optional for kernel 7.x and higher)
- /usr/sap/tmp — the directory in which the SAP operating system collector keeps monitoring data of the local operating system
- /usr/sap/hostctrl — the directory in which SAP control services for the local host are kept (kernel 7.x and higher)

- /etc/cmcluster — the directory in which Serviceguard keeps its legacy configuration files and the node specific package runtime directories
- Local database client software needs to be stored locally on each node. Details can be found in the database sections below.

Part of the content of the local group of directories must be synchronized manually between all nodes of the cluster.

SAP instance (startup) profile names contain either local hostnames or virtual hostnames. SGeSAP will always prefer profiles that use local hostnames to allow individual startup profiles for each host, which might be useful if the failover hardware differs in size.

In clustered SAP environments prior to 7.x releases it is required to install local executables. Local executables help to prevent several causes for package startup or package shutdown hangs due to the unavailability of the centralized executable directory. Availability of executables delivered with packaged SAP components is mandatory for proper package operation. Experience has shown that it is a good practice to create local copies for all files in the central executable directory. This includes shared libraries delivered by SAP.

To automatically synchronize local copies of the executables, SAP components deliver the sapcpe mechanism. With every startup of the instance, sapcpe matches new executables stored centrally with those stored locally.

Directories that Reside on Shared Disks

Volume groups on a SAN shard storage get configured as part of the SGeSAP package.

Instance specific volume groups are required by only one SAP instance or one database instance. They usually get included with exactly the package that is set up for this instance. In this configuration option the instance specific volume groups are included in package.

System specific volume groups get accessed from all instances that belong to a particular SAP System. Environment specific volume groups get accessed from all instances that belong to any SAP System installed in the whole SAP scenario. System and environment specific volume groups should be set up using HA NFS to provide access capabilities to SAP instances on nodes outside of the cluster. The cross-mounting concept of option 1 is not required.

A valuable naming convention for most of these shared volume groups is vg<INSTNAME><SID> or alternatively vg<INSTNAME><SID><INR>. Table 2-4 provide an overview of SAP shared storage for this special setup and maps them to the component and package type for which they occur.

Table 2-4 File systems for the SGeSAP package in NFS Idle Standby Clusters

Mount Point	Access Point	Remarks	VG Name	Device minor number
/sapmnt/<SID>	shared disk and HA NFS	required		
/usr/sap/<SID>	shared disk			
/usr/sap/trans	shared disk and HA NFS	optional		

The table can be used to document used device minor numbers. The device minor numbers of logical volumes need to be identical for each distributed volume group across all cluster nodes.

If you have more than one system, place/usr/sap/put on separate volume groups created on shared drives. The directory should not be added to any package. This ensures that they are independent from any SAP WAS system and you can mount them on any host by hand if needed.

Option 3: SGeSAP CFS Cluster

SGeSAP supports the use of HP Serviceguard Cluster File System for concurrent shared access. CFS is available with selected HP Serviceguard Storage Management Suite bundles. CFS replaces NFS technology for all SAP related file systems. All related instances need to run on cluster nodes to have access to the shared files.

SAP related file systems that reside on CFS are accessible from all nodes in the cluster. Concurrent reads or writes are handled by the CFS layer. Each required CFS disk group and each required CFS mount point

requires a Serviceguard multi-node package. SGeSAP packages are Serviceguard single-node packages. Thus, a package can not combine SGeSAP and CFS related functionality.

Common Directories that are Kept Local

Most common file systems reside on CFS, but there are some directories and files that are kept local on each node of the cluster:

- `/etc/cmcluster` — the directory in which Serviceguard keeps its configuration files and the node specific package runtime directories.
- `/home/<SID>adm` — the home directory of the SAP system administrator with node specific startup log files.
- `/usr/sap/tmp` — the directory in which the SAP operating system collector keeps monitoring data of the local operating system.
- `/usr/sap/<SID>/SYS/exe/run` — optional directory for usage with sapcpe, i.e. a local copy of executables (optional for kernel 7.x and higher).
- `/usr/sap/hostctrl` — the directory in which SAP control services for the local host are kept (kernel 7.x and higher)
- Depending on database vendor and version, it might in addition be required to keep local database client software. Details can be found in the database sections below.

Content of the local group of directories must be synchronized manually between all nodes of the cluster. An exception is the optional local directory for SAP executables `/usr/sap/<SID>/SYS/exe/run`. It gets automatically synchronized as part of every instance startup and shutdown operation. A symbolic link called`/usr/sap/<SID>/SYS/exe/ctrun` needs to be created to access the centrally shared location.

Directories that Reside on CFS

The following table shows a recommended example on how to design SAP file systems for CFS shared access.

Table 2-5 File System Layout for SGeSAP CFS Clusters

Mount Point	Access Point	Package Name Example	DG Name
`/sapmnt/<SID>`	shared disk and CFS	`SG-CFS-DG-<NR1>SG-CFS-MP-<NR1>`	
`/usr/sap/trans`		`SG-CFS-DG-<NR2>SG-CFS-MP-<NR2>`	
`/usr/sap/<SID>`		`SG-CFS-DG-<NR1>SG-CFS-MP-<NR3>`	

The table can be used to document used volume or disk groups across all cluster nodes.

The `/usr/sap/tmp` might or might not be part of the local root file system. This is the working directory of the operating system collector process `saposcol`. The size of this directory will rarely be beyond a few Megabytes.

Database Instance Storage Considerations

SGeSAP internally supports clustering of database technologies of different vendors. The vendors have implemented individual database architectures. The storage layout for SGeSAP cluster environments needs to be discussed individually for each:

- Oracle Single Instance RDBMS
- Oracle Real Application Clusters
- MAXDB Storage Considerations

Table 2-6 Availability of SGeSAP Storage Layout Options for Different Database RDBMS

DB Technology	Supported Platforms	SGeSAP Storage Layout Options	Cluster Software Bundles
Oracle Single-Instance	PA 9000 Itanium		1. Serviceguard or any Serviceguard Storage Management bundle (for Oracle) 2. SGeSAP 3. Serviceguard HA NFS Toolkit
		idle standby	1. Serviceguard 2. SGeSAP 3. Serviceguard HA NFS Toolkit (opt.)
		CFS	1. Serviceguard Cluster File System (for Oracle) 2. SGeSAP
Oracle Real Application Cluster		CFS	1. Serviceguard Cluster File System for RAC 2. SGeSAP
SAPDB MAXDB		NFS	1. Serviceguard or any Serviceguard Storage Management bundle 2. SGeSAP 3. Serviceguard HA NFS Toolkit

Oracle Single Instance RDBMS

Single Instance Oracle databases can be used with all three SGeSAP storage layout options. The setup for NFS and NFS Idle Standby Clusters are identical.

Oracle databases in SGeSAP NFS and NFS Idle Standby Clusters

Oracle server directories reside below `/oracle/<DBSID>`. These directories get shared via the database package

In addition, any SAP Application Server needs access to the Oracle client libraries, including the Oracle National Language Support files (NLS) shown in Table 2-7 NLS Files Default Location. The default location to which the client NLS files get installed differs with the SAP kernel release used:

Table 2-7 NLS Files - Default Location

Kernel Version	Client NLS Location
<=4.6	`$ORACLE_HOME/ocommon/NLS[_<nls_version>]/admin/data`
4.6	`/oracle/<rdbms_version>/ocommon/nls/admin/data`
6.x, 7.x	`/oracle/client/<rdbms_version>/ocommon/nls/admin/data`

It is important to notice, that there always is a second type of NLS directory, called the "server" NLS directory. It gets created during database or SAP Central System installations. The location of the server NLS files is identical for all SAP kernel versions:

`$ORACLE_HOME/common/nls/admin/data`

The setting of the `ORA_NLS[<nls_version>]` variable in the environments of `<sid>adm and ora<sid>` determines whether the client or the server path to NLS is used. The variable gets defined in the `.dbenv_<hostname>.[c]sh` files in the home directories of these users.

During SGeSAP installation it is necessary to create local copies of the client NLS files on each host to which a failover could take place. SAP Central Instances use the server path to NLS files, while Application Server Instances use the client path.

Sometimes a single host may have an installation of both a Central Instance and an additional Application Server of the same SAP System. These instances need to share the same environment settings. SAP recommends using the server path to NLS files for both instances in this case. This won't work with SGeSAP since switching the database would leave the application server without NLS file access.

Oracle 9.x releases no longer maintain NLS compatibility with Oracle 8.x. Also, Oracle 9.x patches introduce incompatibilities with older Oracle 9.x NLS files. The following constraints need to be met:

1. The Oracle RDBMS and database tools rely on an `ORA_NLS[<nls_version>]` setting that refers to NLS files that are compatible to the version of the RDBMS. Oracle 9.x needs NLS files as delivered with Oracle 9.x.
2. The SAP executables rely on an `ORA_NLS[<nls_version>]` setting that refers to NLS files of the same versions as those that were used during kernel link time by SAP development. This is not necessarily identical to the installed database release.

The Oracle database server and SAP server might need different types of NLS files. The server NLS files are part of the database Serviceguard package. The client NLS files are installed locally on all hosts. Special care has to be taken to not mix the access paths for ORACLE server and client processes.

The discussion of NLS files has no impact on the treatment of other parts of the ORACLE client files. The following directories need to exist locally on all hosts on which an Application Server might run. They can not be relocated to different paths. The content needs to be identical to the content of the corresponding directories that are shared as part of the database SGeSAP package. The setup for these directories follows the "on top" mount approach, i.e., the directories might become hidden beneath identical copies that are part of the package:

`$ORACLE_HOME/rdbms/mesg`

`$ORACLE_HOME/oracore/zoneinfo`

`$ORACLE_HOME/network/admin`

Table 2-8 File System Layout for NFS-based Oracle Clusters

Mount Point	Access Point	Potential Owning Packages	VG Type	Volume Group Name	Device Minor Number

`$ORACLE_HOME`
`/oracle/<SID>/saparch`
`/oracle/<SID>/sapreorg`
`/oracle/<SID>/sapdata1`
...
`/oracle/<SID>/sapdatan`
`/oracle/<SID>/origlogA`
`/oracle/<SID>/origlogB`
`/oracle/<SID>/mirrlogA`
`/oracle/<SID>/mirrlogB`
	shared disk	`dbci<DBSID>`
		`jdb"<DBSID"`
		`jdbjci<DBSID>`
		`dbcijci<DBSID>`
			db instance specific		

`/oracle/client`	local	none	environment specific		
some local Oracle client files reside in `/oracle/<SID>` as part of the root filesystem	local	none	db instance specific		

Oracle Real Application Clusters

Oracle Real Application Clusters (RAC) is an option to the Single Instance Oracle Database Enterprise Edition. Oracle RAC is a cluster database with shared cache architecture. The SAP certified solution is based on HP Serviceguard Cluster File System for RAC.

Handling of a RAC database is not included in SGeSAP itself. RAC databases are treated by SGeRAC and Oracle tools, which integrate with SGeSAP. The configuration of SGeSAP packages for non-database components is identical to non-RAC environments.

NOTE: The configurations are designed to be compliant to SAP OSS note 830982. The note describes SAP recommendations for Oracle RAC configurations. A support statement from SAP regarding RAC clusters on HP-UX can be found as part of SAP OSS note 527843. A current support statement in note 527843 is required, before any of the described RAC options can be implemented. The note maintenance is done by

SAP and the note content may change at any time without further notice. Described options may have "Controlled Availability" status at SAP.

Real Application Clusters requires concurrent shared access to Oracle files from all cluster nodes. This can be achieved by installing the Oracle software on Cluster File Systems provided by HP Serviceguard Cluster File System for RAC.

There are node specific files and directories, such as the TNS configuration. These files and directories are copied to a private file system on each node. The node specific files and directories are then removed from the shared disks and symbolic links of the same name are created, the targets of which are the corresponding files in the private file system.

Table 2-9 File System Layout for Oracle RAC in SGeSAP CFS Cluster

Mount Point	Access Point	Potential Owning Packages

`$ORACLE_HOME/oracle/client`
`/oracle/<SID>/oraarch`
`/oracle/<SID>/sapraw`
`/oracle/<SID>/saparch`
`/oracle/<SID>/sapbackup`
`/oracle/<SID>/sapcheck`
`/oracle/<SID>/sapreorg`
`/oracle/<SID>/saptrace`
`/oracle/<SID>/sapdata1...`
`/oracle/<SID>/sapdatan`
`/oracle/<SID>/origlogA`
`/oracle/<SID>/origlogB`
`/oracle/<SID>/mirrlogA`
`/oracle/<SID>/mirrlogB`
	shared disk and CFS	

`tnsnames.ora`	local disk; access via symbolic link from shared disk	

MAXDB Storage Considerations

SGeSAP supports failover of MAXDB databases as part of SGeSAP NFS cluster option

Cluster File Systems can not be used for the MAXDB part of SGeSAP clusters.

The considerations given below for MAXDB will also apply to liveCache and SAPDB clusters unless otherwise noticed.

MAXDB distinguishes an instance dependant path `/sapdb/<DBSID>` and two instance independent paths, called `IndepData` and `IndepPrograms`. By default all three point to a directory below `/sapdb`.

The paths can be configured in a configuration file called `/var/spool/sql/ini/SAP_DBTech.ini`. Depending on the version of the MAXDB database this file contains different sections and settings.

A sample `SAP_DBTech.ini` for a host with a SAPDB 7.4 (LC1) and an APO 3.1 using a SAPDB 7.3 database instance (AP1):

```
[Globals]
IndepData=/sapdb/data
IndepPrograms=/sapdb/programs
[Installations]
/sapdb/LC1/db=7.4.2.3,/sapdb/LC1/db
/sapdb/AP1/db=7.3.0.15,/sapdb/AP1/db
[Databases]
.SAPDBLC=/sapdb/LC1/db
LC1=/sapdb/LC1/db
_SAPDBAP=/sapdb/AP1/db
```

```
AP1=/sapdb/AP1/db
[Runtime]
/sapdb/programs/runtime/7240=7.2.4.0,
/sapdb/programs/runtime/7250=7.2.5.0,
/sapdb/programs/runtime/7300=7.3.0.0,
/sapdb/programs/runtime/7301=7.3.1.0,
/sapdb/programs/runtime/7401=7.4.1.0,
/sapdb/programs/runtime/7402=7.4.2.0,
```

For MAXDB and liveCache Version 7.5 (or higher) the `SAP_DBTech.ini` file does not contain sections `[Installations]` , `[Databases]` and `[Runtime]`. These sections are stored in separate files `Installations.ini`, `Databases.ini` and `Runtimes.ini` in the `IndepData` path `/sapdb/data/config`.

A sample SAP_DBTech.ini, Installations.ini, Databases.ini and Runtimes.ini for a host with a liveCache 7.5 (LC2) and an APO 4.1 using a MAXDB 7.5 (AP2):from `/var/spool/sql/ini/SAP_DBTech.ini`:

```
[Globals]
IndepData=/sapdb/data
IndepPrograms=/sapdb/programs
from /sapdb/data/config/Installations.ini:
[Installations]
/sapdb/LC2/db=7.5.0.15,/sapdb/LC2/db
/sapdb/AP2/db=7.5.0.21,/sapdb/AP2/db
from /sapdb/data/config/Databases.ini:
[Databases]
.M750015=/sapdb/LC2/db
LC2=/sapdb/LC2/db.
M750021=/sapdb/AP2/db
AP2=/sapdb/AP2/db
from /sapdb/data/config/Runtimes.ini:
[Runtime]
/sapdb/programs/runtime/7500=7.5.0.0
```

NOTE: The `[Globals]` section is commonly shared between LC1/LC2 and AP1/AP2. This prevents setups that keep the directories of LC1 and AP1 completely separated.

The following directories are of special interest:

- `/sapdb/programs`: this can be seen as a central directory with all MAXDB executables. The directory is shared between all MAXDB instances that reside on the same host. It is also possible to share the directory across hosts. But it is not possible to use different executable directories for two MAXDB instances on the same host. Furthermore, it might happen that different SAPDB versions get installed on the same host. The files in `/sapdb/programs` have to be of the newest version that any MAXDB on the cluster nodes has. Files in `/sapdb/programs` are downwards compatible. For liveCache 7.4 and APO 3.1 using SAPDB 7.3 this means that within `/sapdb/programs`there have to be the SAPDB 7.4 version executables installed. It is important to realize, that also any SAPDB based SAP application server instance will use this path to access the database client files.
- `/sapdb/data/config`: This directory is also shared between instances, though you can find lots of files that are Instance specific in here, e.g. `/sapdb/data/config/<DBSID>.*` According to SAP this path setting is static.
- `/sapdb/data/wrk`: The working directory of the main MAXDB processes is also a subdirectory of the IndepData path for non-HA setups. If a SAPDB restarts after a crash, it copies important files from

this directory to a backup location. This information is then used to determine the reason of the crash. In HA scenarios, for SAPDB/MAXDB lower that version 7.6, this directory should move with the package. Therefore, SAP provided a way to redefine this path for each SAPBDB/MAXDB individually. SGeSAP expects the work directory to be part of the database package. The mount point moves from `/sapdb/data/wrk` to `/sapdb/data/<DBSID>/wrk` for the clustered setup. This directory should not be mixed up with the directory `/sapdb/data/<DBSID>/db/wrk` that might also exist. Core files of the kernel processes are written into the working directory. These core files have file sizes of several Gigabytes. Sufficient free space needs to be configured for the shared logical volume to allow core dumps.

NOTE: For MAXDB RDBMS starting with version 7.6 these limitations do not exist any more. The working directory is utilized by all instances (`IndepData/wrk`) and can be globally shared.

- `/var/spool/sql`: This directory hosts local runtime data of all locally running MAXDB instances. Most of the data in this directory would become meaningless in the context of a different host after failover. The only critical portion that still has to be accessible after failover is the initialization data in `/var/spool/sql/ini`. This directory is usually very small (< 1Megabyte). With MAXDB and liveCache 7.5 or higher, the only local files are contained in `/var/spool/sql/ini`, other paths are just links to local runtime data in `IndepData` path:

```
dbspeed -> /sapdb/data/dbspeed
diag -> /sapdb/data/diag
fifo -> /sapdb/data/fifo
ipc -> /sapdb/data/ipc
pid -> /sapdb/data/pid
pipe -> /sapdb/data/pipe
ppid -> /sapdb/data/ppid
```

The links need to exist on every possible failover node the MAXDB or liveCache instance is able to run:

- `/etc/opt/sdb`: Only existent when using MAXDB or liveCache 7.5, needs to be local on each node together with entries in `/etc/passwd` and `/etc/group`.

Table 2-10 shows the file system layout for SAPDB clusters.

NOTE: In HA scenarios, valid for SAPDB/MAXDB versions up to 7.6, the runtime directory `/sapdb/data/wrk` is configured to be located at `/sapdb/<DBSID>/wrk` to support consolidated failover environments with several MAXDB instances. The local directory `/sapdb/data/wrk` though is still referred to by the VSERVER processes (`vserver`, `niserver`), which means VSERVER core dump and log files will be located there.

Table 2-10 File System Layout for SAPDB Clusters

Mount Point	Access Point	Potential Owning Packages	VG Type	VG Name	Device Minor Number
/sapdb/<DBSID> /sapdb/<DBSID>/wrk* /sapdb/<DBSID>/data<nr> /sapdb/<dbsid>/saplog<nr>	shared disk	db<DBSID> dbci<DBSID> jdb<dbsid> jdbjci<DBSID>	database specific		
/export/sapdb/programs /export/sapdb/data /export/var/spool/sql/ini	shared disk and HA NFS	db<DBSID> dbci<DBSID> jdb<DBSID> jdbjci<DBSID> SAPNFS	environment specific		
/etc/opt/sdb	local	none			
* only valid for SAPDB/MAXDB versions lower than 7.6					

NOTE: Using tar or cpio is not a safe method to copy or move directories to shared volumes. In certain circumstances file or ownership permissions may not correctly transported, especially files having the s-bit set: `/sapdb/<SID>/db/pgm/lserver` and `/sapdb/<SID>/db/pgm/dbmsrv`. These files are important for the vserver process ownership and they have an impact on starting the SAPDB via `<SID>adm`. These files should retain the same ownership and permission settings after moving them to a shared volume.

Database and SAP instances depend on the availability of `/sapdb/programs`. To minimize dependencies between otherwise unrelated systems, it is strongly recommended to use a dedicated SAPNFS package, especially in case that the cluster has additional SAP application servers installed, more than one SAPDB is installed or the database is configured in a separate DB package. Keeping local copies is possible, though not recommended because of the fact that there are no administration tools that keep track of the consistency between the local copies of these files on all the systems.

3 Step-by-Step Cluster Conversion

This chapter describes in detail how to implement a SAP cluster using Serviceguard and Serviceguard Extension for SAP (SGeSAP). It is written in the format of a step-by-step guide. It gives examples for each task in great detail. Actual implementations might require a slightly different approach. Many steps synchronize cluster host configurations or virtualize SAP instances manually. If these tasks are already covered by different means, it might be sufficient to quickly check that the requested result is already achieved. For example, if the SAP application component was already installed using a virtual IP address, many steps from the SAP Application Server configuration section can be omitted.

Various Serviceguard modules are available with SGeSAP, including

- `sgesap/sapinstance` for clustering of one or more SAP instances, for example SAP Central Instances, System Central Services, Replication Instances, ABAP Application Servers and JAVA Application Servers of a single SAP system
- `sgesap/dbinstance` for ORACLE or MAXDB RDBMS
- `sgesap/sapextinstance` for the handling of non-clustered SAP software
- `sgesap/sapinfra` for clustering of SAP infrastructure software

The generic SGeSAP package module can be referred to as `sgesap/sapinstance`. It can be used to add one or more Netweaver instances of a single SAP system to a Serviceguard package. `sgesap/scs` can be used to primarily add only the Netweaver software single points of failure, ie. Central Instances or System Central Service Instances. `sgesap/ers` can be used to primarily add only the Enqueue Replication Service instances that correspond to System Central Service instances.

Various SGeSAP legacy package types are supported, including

- SAP Central Instances and ABAP System Central Services (ci),
- Java System Central Services (`jci`),
- Database Instances for ABAP components (`db`),
- Database Instances for J2EE components (`jdb`),
- Replicated Enqueue for both ABAP and JAVA System Central Services (`[a]rep`),
- ABAP Dialog Instances and Application Servers (`d`),
- JAVA Dialog Instances and Application Servers (`jd`)
- Highly Available NFS (`sapnfs`) and
- Combinations of these package types (`dbci, jdbjci, dbcijci`,...).

Refer to the support matrix published in the release notes of SGeSAP to get details whether your SAP application component is supported in Serviceguard clusters on HP-UX.

liveCache legacy packages require legacy package type lc. liveCache modules are designed with module `sgesap/livecache`. MDM can be clustered with a single legacy package `mgroup`, or with a set of five packages `mmaster, mds, mdis, mdss, mdb`. Refer to chapter 4 for Livecache package types, and Chapter 5 for MDM legacy package types.

In order to ensure a consistent naming scheme in the following installation and configuration steps, the SGeSAP package names are assembled from the package type and SAP System ID:

`<pkg_name> = <pkg_type><SID>` or, in case of ambiguous results

`<pg_name> = <pkg_type><INSTNR><SID>`

Examples: dbC11, ciC11, dbciC11, dbcijciC11, jdbC11, jciC11, jdbjciC11, d01C11, d02C11, ers03C11

There is one exception in the naming convention, which concerns the dedicated NFS package sapnfs that might serve more than one SAP SID.

SGeSAP modules are implemented to work independent of the used package naming. For these packages, the above naming scheme is a recommendation.

For a description and combination restrictions on those packages, refer to chapter 1 - Designing SGeSAP Cluster Scenarios.

The legacy package installation steps cover HP-UX 11i v1, HP-UX 11i v2 and HP-UX 11i v3 using Serviceguard 11.16 or higher.

Modular packages can be used with HP-UX 11i v2 and HP-UX 11i v3 using Serviceguard 11.18 or higher.

The package creation process is split into the following logical tasks:

- SAP Preparation
- HP-UX Configuration
- Modular Package Configuration
- Legacy Package Configuration
- HA NFS Toolkit Configuration
- Auto FS Configuration
- Database Configuration
- SAP Application Server Configuration

The tasks are presented as a sequence of steps. Each installation step is accompanied by a unique number of the format XXnnn, where nnn are incrementing values and XX indicates the step relationship, as follows:

- ISnnn—Installation Steps mandatory for all SAP packages
- OSnnn—Optional Steps
- LSnnn—Installation Steps mandatory for legacy packages only
- MSnnn—Installation Steps mandatory for module based packages only
- ORnnn—ORacle database only steps
- SDnnn—SAPDB/MAXDB or LC database only steps
- REnnn—Replicated Enqueue only steps

If you need assistance during the installation process and need to contact HP support, you can refer to the installation step numbers to specify your issue. Also include the version of this document. The installation step numbers were reorganized for this document version and don't resemble to earlier releases.

Whenever appropriate, HP-UX sample commands are given to guide you through the integration process. It is assumed that hardware as well as the operating system and Serviceguard are already installed properly on all cluster hosts. Sometimes a condition is specified with the installation step. Follow the information presented only if the condition is true for your situation.

NOTE: For installation steps in this chapter that require the adjustment of SAP specific parameter in order to run the SAP system in a switchover environment example values are given. These values are for reference ONLY and it is recommended to read and follow the appropriate SAP OSS notes for SAP's latest recommendation. Whenever possible the SAP OSS note number is given.

The SAP Application Server installation types are ABAP-only, Java-only and Add-in. The latter includes both the ABAP and the Java stack.

In principle all SAP cluster installations look very similar. Older SAP systems get installed in the same way as they would without a cluster. Cluster conversion takes place afterwards and includes a set of manual steps. Some of these steps can be omitted since the introduction of high availability installation options to the SAP installer SAPINST. In this case, a part of the cluster configuration is done prior to the SAP installation as such. The SAP Instances can then be installed into a virtualized environment, which obsoletes the SAP Application Server Configuration steps that usually concluded a manual cluster conversion.

Therefore, it is important to first decide which kind of SAP installation is intended. The installation of a SAP High Availability System was introduced with Netweaver 2004s. For Netweaver 2004 JAVA-only installations there is a similar High Availability Option for SAPINST. All older SAP kernels need to be clustered manually. The SAP preparation chapter covers all three cases. It also describes how Enqueue Replication can be activated for use with SGeSAP. The exact steps for that also depend on the SAP release that gets used.

Other differences in cluster layout derive from the usage of HP Serviceguard Cluster Filesystem. Alternative setups use HA NFS packages and automounter technology to ensure cluster-wide access to filesystems.

Finally, the underlying database of course also causes slightly different installation steps.

SAP Preparation

This section covers the SAP specific preparation, installation and configuration before creating a high available SAP System landscape. This includes the following logical tasks:

- SAP Pre-Installation Considerations
- Replicated Enqueue Conversion

SAP Pre-Installation Considerations

This section gives additional information that help with the task of performing SAP installations in HP Serviceguard clusters. It is not intended to replace any SAP installation manual. SAP installation instructions provide complementary information and should be consulted in addition to this.

SAP Netweaver 2004s introduced High Availability installation options. In combination with SGeSAP they can also be activated to guide the clustering of SAP JAVA-only applications based on kernel 6.40. The SAP Enterprise Portal 6.0 belongs to this group of applications. All SAP components that are based on earlier SAP technology should to be installed in the standard way as described by the SAP Installation Guides. They can be clustered after the initial installation. In any case, it makes sense to already set up required file systems as documented in Chapter 2 'Planning the Storage Layout' to prevent avoidable conversion activities in a later stage.

The following paragraphs are divided into these installation options:

- SAP Netweaver High Availability
- Generic SAP Installation

SAP Netweaver High Availability

The SAP Netweaver High Availability options were officially introduced with Netweaver 2004s technology. They are based on the SAP Application Server 7.x. The SAPINST installer for SAP kernel 7.x offers the following installation options out of the box:

- Central System
- Distributed System
- High Availability System

The Central System and Distributed System installations build a traditional SAP landscape. They will install a database and a monolithic Central Instance. Exceptions are Java-only based installations.

NOTE: For Java-only based installations the only possible installation option is a High Availability System installation.

It is strongly recommended to use the "High Availability System" option for all new installations that are meant to be used with SGeSAP.

A SAP Application Server 7.0 may consist of any combination of the following components:

- Java System Central Services Instance (SCS) [Java Message and Enqueue Server]
- ABAP System Central Services Instance (ASCS) [ABAP Message and Enqueue Server]
- Dialog Instance(s) (D, JD, DVEBMGS) [ABAP and/or Java stack]

The potential SPOFs are the SCS, ASCS and DB instances. The Dialog instance can be installed redundantly on nodes inside or outside the cluster. The ABAP DVEBMGS instance for Netweaver 2004s (or higher) is similar to a simple Dialog Instance, except that it is preconfigured to contain the services Batch, Update, Spool and Gateway. For JAVA, a Central Instance runs the Java Software Deployment Manager (SDM). These services though can be also configured redundantly with other Dialog Instances.

The SAP Netweaver CDs/DVDs must be available either as physical copies or images on the local or shared file system for the duration of the installation.

As preparation, simple Serviceguard packages for the clustered instances have to be created. They provide the virtual IP addresses that are required during installation. The package(s) will later on be altered to utilize

SGeSAP functionality. It will be more convenient to do this once the SAP installation has taken place. The following steps are performed as root user to prepare the cluster for the SAP installation.

Preparation steps MS12xx should be followed to create module-based packages. Preparation steps LS12xx should be followed to create legacy packages.

Preparation Step: MS1200

Create a tentative Serviceguard package configuration for one or more SAP instances.

The first thing required is a SGeSAP configuration file that can be used to define the SAP Serviceguard failover package. It is usually created with the following command:

```
cmmakepkg -m sgesap/sapinstance [-m ...] >/sap.config
```

For a database package, the module `sgesap/dbinstance` can be specified.

NOTE: SGeSAP modules can be referred to by using SGeSAP legacy package types. Covered package types include: `ci, scs` and `ascs`, `rep` and `arep`.

Specification examples for SGeSAP modules:

```
cmmakepkg -m sgesap/db -m sgesap/ci
```

creates a single package configuration file for database and SAP instance(s) (one package concept)

```
cmmakepkg -m sgesap/db
cmmakepkg -m sgesap/ci
```

separates database and SAP instances into two package configuration files (two package concept)

```
cmmakepkg -m sgesap/scs
cmmakepkg -m sgesap/ers
```

separates System Central Service and Enqueue Replication into two packages

```
cmmakepkg -m sgesap/scs -m sgesap/ers
```

would immediately issue an error message, because System Central Services and Enqueue Replication cannot share the same package.

Preparation Step: MS1210

The created configuration file needs to be edited.

Refer to the Managing Serviceguard user's guide for general information about the generic file content. A minimum configuration will do for the purpose of supporting the SAP installation. At least the following parameters should be edited: `package_name, node_name, ip_address` and `monitored_subnet`.

The `package_name` can be chosen freely. It is often a good approach to stick to the naming convention that combines the name of the SAP Instance type and the SAP System ID. Examples: dbciC11, scsC11. Specify `node_name` entries for all hosts on which the package should be able to run. There is a section that defines a virtual IP address array. All virtual IP addresses specified here will become associated to the SAP and database instances that are going to be installed. Specify at least one virtual IP. Specify subnets to be monitored in the `monitored_subnet` section.

The only SAP specific parameters that needs to be set is the planned SAP system id `sap_system`.

Preparation Step: MS1220

Create a debug file on the system on which the installation takes place.

The debug file allows manual SAP instance shutdown and startup operations during installation.

```
touch /var/adm/cmcluster/debug_<package_name>
```

It does not matter, if the system is meant to be run in a multi-tier fashion that separates the database from the ASCS instance by running them on different cluster nodes during normal operation. For convenience, all installation steps should be done on a single machine. Due to the virtualization, it is easy to separate the instances later on.

Preparation Step: MS1230

The configuration file needs to be applied and the packages started.

This step assumes that the cluster as such is already configured and started. Please refer to the Managing Serviceguard user's guide if more details are required.

cmapplyconf -P ./sap.config

cmrunpkg -n <sap_installation_host> <package_name>

All virtual IP addresses should now be configured. A ping command should reveal that they respond to communication requests.

Preparation Step: IS1300

Before installing the SAP Application Server 7.0 some OS-specific parameters have to be adjusted.

Verify or modify HP-UX kernel parameters as recommended by the SAP Master Guide Part 1. Be sure to propagate changes to all nodes in the cluster.

The SAP Installer checks the OS parameter settings with a tool called "Prerequisite Checker" and stops the installer when the requirements are not met.

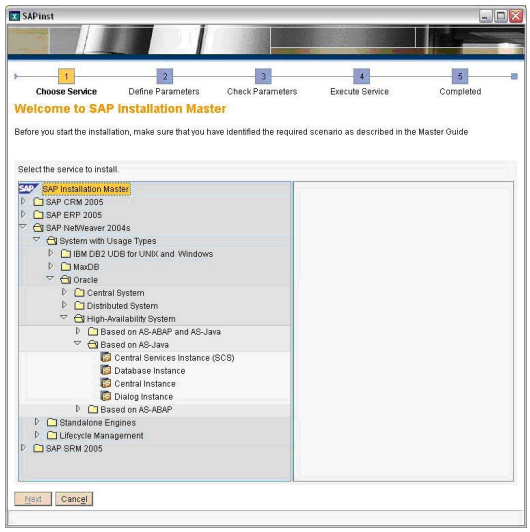
Check the installed Java SDK with the requirement as of the SAP Master Guide Part 1. Be sure to install the required SDK on all nodes in the cluster.

In case the J2EE engine uses security features that utilize the need for the Java Cryptographic Toolkit, it has to be downloaded both from the Sun website as well as from the SAP Service Marketplace.

Preparation Step: IS1320

Before invoking the SAP installer, some additional requirements have to be met that are described in the SAP installation documentation.

These are not specific to cluster implementations and apply to any SAP installation. They usually involve the setting of environment variables for SAPINST, like the DISPLAY variable, the temporary installation directory TMP, etc.



Installation Step: IS1330

The installation is done using the virtual IP provided by the Serviceguard package. SAPINST can be invoked with a special parameter called SAPINST_USE_HOSTNAME.

This prevents the installer routines from comparing the physical hostname with the virtual address and drawing wrong conclusions.

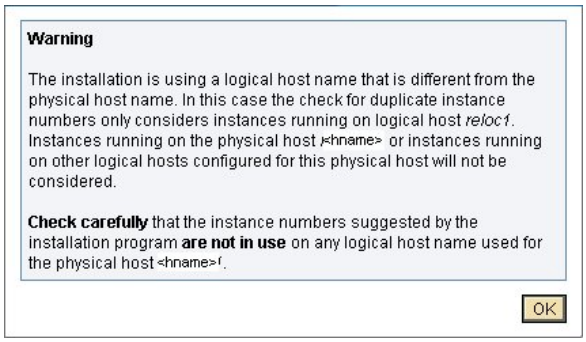
The installation of the entire SAP Application Server 7.0 will happen in several steps, depending on the installation type. Each time a different virtual hostname can be provided.

- First, the SAP System Central Services Component (SCS and/or ASCS) will be installed using the virtual IP contained in the corresponding Serviceguard package.
- Then, the database instance will be set up and installed on the relocatable IP address of the DB package.
- After that, the Central Instance and all required Dialog Instances are established. Virtual IPs can be used here, too. It is recommended to do this, because it preserves the flexibility to move instances

between physical machines. HP does not support a conversion to a virtual IP after the initial installation on a physical hostname for SAP JAVA engines.

The SAPINST_USE_HOSTNAME option can be set as an environment variable, using export or setenv commands. It can also be passed to SAPINST as an argument.

```
cd <COMPONENT_TMP_DIR>
<SAPINST_MASTER_DVD>/IM<X>_OS/SAPINST/UNIX/<OS>/sapinst \
SAPINST_USE_HOSTNAME=<reloc_jci>
```



NOTE: The SAPINST_USE_HOSTNAME parameter is not mentioned in SAP installation documents for the SAP JAVA engine 6.40, but it is officially supported. The installer will show a warning message that has to be confirmed.

When starting the SAPINST installer for kernel 6.40, the first screen shows installation options that are generated from an XML file called product.catalog located at <SAPINST_MASTER_DVD>/IM<X>_OS/SAPINST/UNIX/<OS>. </OS></X></SAPINST_MASTER_DVD>The standard catalog file product.catalog has to be either:

- replaced by product_ha.catalog in the same directory on a local copy of the DVD or
- the file product_ha.catalog can be passed as an argument to the SAPINST installer

It is recommended to pass the catalog as an argument to SAPINST. The XML file that is meant to be used with SGeSAP clusters is included on the installation DVD/CD's distributed by SAP.

The SAPINST_USE_HOSTNAME option can be set as an environment variable, using export or setenv commands. It can also be passed to SAPINST as an argument.

```
cd <COMPONENT_TMP_DIR>
<SAPINST_MASTER_DVD>/IM<X>_OS/SAPINST/UNIX/<OS>/sapinst \
<SAPINST_MASTER_DVD>/IM<X>_OS/SAPINST/UNIX/<OS>\
product_ha.catalog\
SAPINST_USE_HOSTNAME=reloc_jci>
```

The SAP installer should now be starting.

Afterwards, the virtualized installation is completed, but the cluster still needs to be configured. The instances are now able to run on the installation host, provided the corresponding Serviceguard packages got started up front. It is not yet possible to move instances to other nodes, monitor the instances or trigger automated failovers. Do not shut down the Serviceguard packages while the instances are running. It is possible to continue installing content for the SAP J2EE Engine, before the cluster conversion as described in the sections below gets performed.

Replicated Enqueue Conversion

This section describes how a SAP ABAP Central Instance DVEBMGS<INSTNR> can be converted to use the Enqueue Replication feature for seamless failover of the Enqueue Service. The whole section can be skipped if Enqueue Replication is not going to be used. It can also be skipped in case Replicated Enqueue is already installed. The proceeding manual conversion steps can be done for SAP applications that are based on ABAP kernel 4.6D and 6.40. These kernels are supported on SGeSAP with Replicated Enqueue, and SAP

is not delivering installation routines that install Replicated Enqueue configurations for these releases, so the manual conversion steps become necessary. The 4.6D kernel does require some kernel executables of the 6.40 kernel to be added.

If the SAP installation was done for Netweaver 2004 Java-only, Netweaver 2004s, or a newer release as documented in section 'SAP Installation Considerations', only the second part 'Creation of Replication Instance' is required. The split of the Central Instance is then already effective and a [A]SCS instance was created during installation.

In this case it is sufficient to ensure that the [A]SCS startup profile does not use local Restart for the enqueue process and that the instance profile contains recommended replication parameter settings, eg:

```
enque/server/internal_replication = true
enque/server/replication = true
enque/server/threadcount = 1
enque/enrep/keepalive_count = 0
enque/process_location = local
enque/table_size = 4096
ipc/shm_psize_34 = 0
```

Using Replicated Enqueue heavily changes the SAP instance landscape and increases the resource demand: Two additional SAP instances will be generated during the splitting procedure. There is a requirement for at least one additional unique SAP System ID. Unique means, that the ID is not in use by any other SAP instance of the cluster. There is also a requirement for one or two additional shared LUNs on the SAN and one or two additional virtual IP addresses for each subnet. The LUNs need to have the size that is required for a SAP Instance directory of the targeted kernel release.

Splitting an ABAP Central Instance

The SPOFs of the DVEBMGS<INSTNR> instance will be isolated in a new instance called ABAP System Central Services Instance ASCS<INSTNR>. This instance will replace DVEBMGS<INSTNR> for the ci package type. The remaining parts of the Central Instance can then be configured as Dialog Instance D<INSTNR_2>. The ASCS Instance should then only be started and stopped with the cluster package startup and halt commands instead of using manual shell operations.

NOTE: The Dialog Instance D<INSTNR_2>that results from the conversion also represents one or more Single Points of Failure for many scenarios. In these cases,D<INSTNR_2> should also be clustered with SGeSAP. It is not even unusual to combine ASCS<INSTNR> andD<INSTNR_2> in a single SGeSAP package. It makes sense, even though the resulting package contains the same components like a traditional package for DVEBMGS<INSTNR> would. Seamless failover with Replicated Enqueue can not be achieved without splitting up DVEBMGS<INSTNR> into two instances.

Logon as root to the server on which the Central Instance DVEBMGS<INSTNR> was installed.

Replicated Enqueue Conversion: RE010

Create a new mountpoint:

```
su - <sid>adm
mkdir /usr/sap/<SID>/ASCS<INSTNR>
```

Replicated Enqueue Conversion: RE020

A volume group needs to be created for the ASCS instance.

The physical device(s) should be created as LUN(s) on shared storage. Storage connectivity is required from all nodes of the cluster that should run the ASCS. For the volume group, one logical volume should get configured. For the required size, refer to the capacity consumption of /usr/sap/<SID>/DVEBMGS<INSTNR>. This should provide a conservative upper limit that leaves reasonable headroom.

Mount the new logical volume to the mountpoint created in RE010.

Replicated Enqueue Conversion: RE030

Create instance subdirectories in the mounted logical volume.

They will be switched between the cluster nodes later.

```
su - <sid>adm
cd /usr/sap/<SID>/ASCS<INSTNR>
mkdir data log sec work
```

Replicated Enqueue Conversion: RE040

If the used SAP kernel has a release older than 6.40...

Download the executables for the Standalone Enqueue server from the SAP Service Marketplace and copy them to `/sapmnt/<SID>/exe`.

There should be at least three files that are added/replaced: `enserver`, `enrepserver` and `ensmon`.

Make sure these files are part of the `sapcpe` mechanism for local executable creation (see Chapter 2).

This step will create a version mix for the executables. The Standalone Enqueue executables get taken from the 6.40 kernel. Special caution has to be taken to not replace them with older releases later on. This might happen by accident when kernel patches get applied.

Replicated Enqueue Conversion: RE050

Create instance profile and startup profile for the ASCS Instance.

These profiles get created as `<sid>adm` in the NFS-shared `/sapmnt/<SID>/profile` directory.

Here is an example template for the instance profile `<SID>_ASCS<INSTNR>_<CIRELOC>`:

```
#-----------------------------------
# general settings
#-----------------------------------
SAPSYSTEMNAME=<SID>
INSTANCE_NAME=ASCS<INSTNR>
SAPSYSTEM=<INSTNR>
SAPLOCALHOST=<CIRELOC>
SAPLOCALHOSTFULL=<CIRELOC>.<domain>
#-----------------------------------
# enqueueserver settings
#-----------------------------------
enque/server/internal_replication = true
enque/server/replication = true
enque/server/threadcount = 1
enque/enrep/keepalive_count = 0
enque/process_location = local
enque/table_size = 4096
ipc/shm_psize_34 = 0
#-----------------------------------
# messageserver settings
#-----------------------------------
rdisp/mshost=<CIRELOC>
#-----------------------------------
# prevent shmem pool creation
#-----------------------------------
ipc/shm_psize_16 = 0
ipc/shm_psize_24 = 0
ipc/shm_psize_34 = 0
ipc/shm_psize_66 = 0
```

This template shows the minimum settings. Scan the old `<SID>_DVEBMGS<INSTNR>_<local_ip>` profile to see whether there are additional parameters that apply to either the Enqueue Service or the Message Service. Individual decisions need to be made whether they should be moved to the new profile.

Here is an example template for the startup profile `START_ASCS<INSTNR>_<CIRELOC>`

```
#---------------------------------------------------------------------
SAPSYSTEMNAME =<SID>
```

```
INSTANCE_NAME =ASCS<INSTNR>
-----------------------------------------------------------------------
# start SCSA handling
#-----------------------------------------------------------------------
Execute_00 =local $(DIR_EXECUTABLE)/sapmscsa -n
               pf=$(DIR_PROFILE)/<SID>_ASCS<INSTNR>_<CIRELOC>
#-----------------------------------------------------------------------
# start message server
#-----------------------------------------------------------------------
MS =ms.sap<SID>_ASCS<INSTNR>
Execute_01 =local rm -f $(_MS)
Execute_02 =local ln -s -f $(DIR_EXECUTABLE)/msg_server $(_MS)
Start_Program_01 =local $(_MS) pf=$(DIR_PROFILE)/<SID>_ASCS<INSTNR>_<CIRELOC>
#-----------------------------------------------------------------------
# start syslog collector daemon
#-----------------------------------------------------------------------
CO =co.sap<SID>_ASCS<INSTNR>
Execute_03 =local rm -f $(_CO)
Execute_04 =local ln -s -f $(DIR_EXECUTABLE)/rslgcoll $(_CO)
Start_Program_02 =local $(_CO) -F pf=$(DIR_PROFILE)/<SID>_ASCS<INSTNR>_<CIRELOC>
#-----------------------------------------------------------------------
# start enqueue server
#-----------------------------------------------------------------------
EN = en.sap<SID>_ASCS<INSTNR>
Execute_05 = local rm -f $(_EN)
Execute_06 = local ln -s -f $(DIR_EXECUTABLE)/enserver $(_EN)
Start_Program_03 = local $(_EN) pf=$(DIR_PROFILE)/<SID>_ASCS<INSTNR>_<CIRELOC>
#-----------------------------------------------------------------------
# start syslog send daemon
#-----------------------------------------------------------------------
SE =se.sap<SID>_ASCS<INSTNR>
Execute_07 =local rm -f $(_SE)
Execute_08 =local ln -s -f $(DIR_EXECUTABLE)/rslgsend $(_SE)
Start_Program_04 =local $(_SE) -F pf=$(DIR_PROFILE)/<SID>_ASCS<INSTNR>_<CIRELOC>
#-----------------------------------------------------------------------
```

Replicated Enqueue Conversion: RE060

Adopt instance profile and startup profile for the *DVEBMGS<INSTNR>* ***instance.***

The goal of this step is to strip the Enqueue and Message Server entries away and create a standard Dialog Instance. A second alteration is the replacement of the Instance Number of the Central Instance which now belongs to ASCS and AREP.

The new Dialog Instance profile `<SID>_DVEBMGS<INSTNR_2>_<DRELOC>` differs from the original `<SID>_DVEBMGS<INSTNR>_<ip_address>` profile in several ways:

All configuration entries for Message and Enqueue Service need to be deleted, e.g.

```
rdisp/wp_no_enq=1
```

must be removed. Several logical names and address references need to reflect a different relocatable address and a different Instance Number. For example:

```
SAPSYSTEM=<INSTNR_2>
```

```
rdisp/vbname = <DRELOC>_<SID>_<INSTNR_2>
```

```
SAPLOCALHOST=<DRELOC>
```

```
SAPLOCALHOSTFULL=<DRELOC>.domain
```

The exact changes depend on the individual appearance of the file for each installation. The startup profile is also individual, but usually can be created similar to the default startup profile of any Dialog Instance. Here is an example template for a startup profile `START_D<INSTNR_2>_<DRELOC>`:

Scan the old <SID>_DVEBMGS<INSTNR>_<local_ip> profile to see whether there are additional parameters that apply to either the Enqueue Service or the Message Service. Individual decisions need to be made whether they should be moved to the new profile.

Here is an example template for the startup profile START_DVEBMGS<INSTNR>_<DRELOC>

```
#-----------------------------------------------------------------------
SAPSYSTEMNAME = <SID>
INSTANCE_NAME = DVEBMGS<INSTNR_2>
#-----------------------------------------------------------------------
# start SCSA
#-----------------------------------------------------------------------
Execute_00 = local $(DIR_EXECUTABLE)/sapmscsa -n
            pf=$(DIR_PROFILE)/<SID>_DVEBMGS<INSTNR_2>_<DRELOC>
#-----------------------------------------------------------------------
# start application server
#-----------------------------------------------------------------------
_DW = dw.sap<SID>_DVEBMGS<INSTNR_2>
Execute_01 = local rm -f $(_DW)
Execute_02 = local ln -s -f $(DIR_EXECUTABLE)/disp+work $(_DW)
Start_Program_01 = local $(_DW) pf=$(DIR_PROFILE)/<SID>_DVEBMGS<INSTNR_2>_<DRELOC>
#-----------------------------------------------------------------------
# start syslog send daemon
#-----------------------------------------------------------------------
_SE = se.sap<SID>_<INSTNR_2>
Execute_03 = local rm -f $(_SE)
Execute_04 = local ln -s -f $(DIR_EXECUTABLE)/rslgsend $(_SE)
Start_Program_02 = local $(_SE) -F pf=$(DIR_PROFILE)/<SID>_DVEBMGS<INSTNR_2>_<DRELOC>
#-----------------------------------------------------------------------
```

Creation of Replication Instance

This section describes how to add Enqueue Replication Services (ERS) to a system that has SAP kernel 4.6, 6.x or 7.0 ASCS and/or SCS instances. The section can be skipped for SAP kernel 7.10 and higher. Use the installation routines for ERS instances as provided by the SAP installer instead.

The ASCS (ABAP) or SCS (JAVA) instance will be accompanied with an ERS instance that permanently keeps a mirror of the [A]SCS internal state and memory. As a result, no information gets lost due to failing SAP System Central Services, if the SGeSAP package that includes [A]SCS fails over to the node that runs ERS during runtime.

NOTE: If both SCS and ASCS are intended to take advantage of Replicated Enqueue, then the following steps need to be taken twice.

Replicated Enqueue Conversion: RE070

Create a new mountpoint.

```
su - <sid>adm
mkdir /usr/sap/<SID>/ERS<INSTNR>
```

Replicated Enqueue Conversion: RE080

A volume group needs to be created for the ERS instance.

The physical device(s) should be created as LUN(s) on shared storage. Storage connectivity is required from all nodes of the cluster that should be able to run the Replicated Enqueue. For the volume group, one logical volume should get configured.

Mount the new logical volume to the mountpoint created in RE070.

NOTE: The <INSTNR> of the ERS has to be different from the instance number of the ASCS and SCS.

Replicated Enqueue Conversion: RE090

Create instance subdirectories in the mounted logical volume

They will be switched between the cluster nodes later.

```
su - <sid>adm
cd /usr/sap/<SID>/ERS<INSTNR>
mkdir data log exe work profile
```

Starting with SAP kernel 7.0, the following subdirectory structure also needs to be created:

```
mkdir -p exe/servicehttp/sapmc
```

Replicated Enqueue Conversion: RE095

A couple of required SAP executables should be copied from the central executable directory *`/sapmnt/<SID>/exe`* ***to the instance executable directory*** *`/usr/sap/<SID>/ERS<INSTNR>/exe`*

For SAP kernel 6.40 the list includes:

`enqt, enrepserver, ensmon, libicudata.so.30, libicui18n.so.30, libicuuc.so.30, libsapu16_mt.so, libsapu16.so, librfcum.so, sapcpe` and `sapstart`.

For some releases, the shared library file extension `.so` is replaced by `.sl`. Apart from this, the procedure remains the same. Copy these files and add a file named `ers.lst` in the instance executable directory. The`ers.lst`should include these lines

```
enqt
enrepserver
ensmon
libicudata.so.30
libicui18n.so.30
libicuuc.so.30
libsapu16_mt.so
libsapu16.so
librfcum.so
sapcpe
sapstart
ers.lst
```

For SAP kernel 7.00 or higher, the following executables need to be copied in addition:

```
sapstartsrv
sapcontrol
servicehttp/sapmc/sapmc.jar
servicehttp/sapmc/sapmc.html
servicehttp/sapmc/frog.jar
servicehttp/sapmc/soapclient.jar
```

For SAP kernel 7.00 or higher, the ers.lst file needs to have additional lines:

```
sapstartsrv
sapcontrol
servicehttp
```

The following script example cperinit.sh performs this step for 7.00 kernels.

```
for i in enqt enrepserver ensmon libicudata.so.30 libicui18n.so.30 libicuuc.so.30 libsapu16_mt.so libsapu16.so librfcum.so sapcpe
 sapstart sapstartsrv sapcontrol
do
  cp /sapmnt/$1/exe/$1 /usr/sap/$1/$2/exe
  echo $i >> /usr/sap/$1/$2/exe/ers.lst
done
echo servicehttp >> /usr/sap/$1/$2/exe/ers.lst
echo ers.lst >> /usr/sap/$1/$2/exe/ers.lst
```

```
for i in sapmc.jar sapmc.html frog.jar soapclient.jar
do
  cp /sapmnt/$1/exe/servicehttp/sapmc/$i /usr/sap/$1/$2/exe/servicehttp/sapmc/$i
done
```

The script needs to be called providing a SAP SID and the instance.

Example: `./cperinit.sh C11 ERS00`

Replicated Enqueue Conversion: RE100

Create instance profile and startup profile for the ERS Instance.

These profiles get created as `<sid>adm` in the instance profile directory- `/usr/sap/<SID>/ERS<INSTNR>/profile`.

Even though this is a different instance, the instance number is identical to the instance number of the [A]SCS. The virtual IP address used needs to be different in any case.

Here is an example template for the instance profile `<SID>_ERS<INSTNR>_<[A]REPRELOC>`:

```
#-------------------------------------------------------------------
# system settings
#-------------------------------------------------------------------
SAPSYSTEM = <INSTNR>
SAPSYSTEMNAME = <SID>
INSTANCE_NAME = ERS<INSTNR>
#-------------------------------------------------------------------
# Special settings for this manually set up instance
#-------------------------------------------------------------------
DIR_EXECUTABLE = $(DIR_INSTANCE)/exe
DIR_PROFILE = $(DIR_INSTANCE)/profile
DIR_CT_RUN = /usr/sap/<SID>/SYS/exe/runU
#-------------------------------------------------------------------
# Settings for enqueue monitoring tools (enqt, ensmon)
#-------------------------------------------------------------------
enque/process_location = REMOTESA
rdisp/enqname = $(rdisp/myname)
#-------------------------------------------------------------------
# standalone enqueue details from (A)SCS instance
#-------------------------------------------------------------------
SCSID = <INR>
SCSHOST = <[J]CIRELOC>
enque/serverinst = $(SCSID)
enque/serverhost = $(SCSHOST)
```

Here is an example template for the startup profile `START_ERS<INSTNR>_<[A]REPRELOC>`:

```
#-------------------------------------------------------------------
SAPSYSTEM = <INSTNR>
SAPSYSTEMNAME = <SID>
INSTANCE_NAME = ERS<INSTNR>
#-------------------------------------------------------------------
# Special settings for this manually set up instance
#-------------------------------------------------------------------
SCSID = <INR>
DIR_EXECUTABLE = $(DIR_INSTANCE)/exe
DIR_PROFILE = $(DIR_INSTANCE)/profile
DIR_CT_RUN = /usr/sap/$(SAPSYSTEMNAME)/SYS/exe/runU
SETENV_00 = PATH=$(DIR_INSTANCE)/exe:%(PATH)
SETENV_01 = LD_LIBRARY_PATH=$(DIR_EXECUTABLE)
_PF = $(DIR_PROFILE)/<SID>_ERS<INSTNR>_<[A]REPRELOC>
#-------------------------------------------------------------------
# Copy SAP Executables
```

```
#--------------------------------------------------------------------
_CPARG0 = list:$(DIR_EXECUTABLE)/ers.lst
Execute_00 = immediate $(DIR_EXECUTABLE)/sapcpe $(_CPARG0) pf=$(_PF)
#--------------------------------------------------------------------
# start enqueue replication server
#--------------------------------------------------------------------
_ER = er.sap$(SAPSYSTEMNAME)_$(INSTANCE_NAME)
Execute_01 = immediate rm -f $(_ER)
Execute_02 = local ln -s -f $(DIR_EXECUTABLE)/enrepserver $(_ER)
Start_Program_00 = local $(_ER) pf=$(_PF) NR=$(SCSID)
```

The DIR_CT_RUN in these profiles should equal the value of DIR_CT_RUN(if specified) or DIR_EXECUTABLE from the [A]SCS instance profile. It is essential that the binaries from [A]SCS and ERS instances are from the same binary set.

Make sure that the [A]SCS instances are not configured for automatic restart in the SAP profile.

HP-UX Configuration

A correct HP-UX configuration ensures that all cluster nodes provide the environment and system configuration required to run SAP Application Server based business scenarios.

Several of the following steps must be repeated on each node. Record the steps completed for each node, as you complete them. This helps identify errors in the event of a malfunction later in the integration process. The HP-UX configuration task is split into the following sections:

- Directory Structure Configuration - This section adds implementation practices to the architectural decisions made in the chapter "Storage Layout Considerations". It gives some guidance how to achieve the differences in directory structure layout for a clustered SAP system as compared to a standard installation.

NOTE: Most of these steps in this section are only performed once on the primary host.

The primary host is where the SAP Instance was installed. If your database is currently running on a machine different from this, the database vendor dependent steps will have to be done on the host on which the database was installed.

- Cluster Node Synchronization - There are a couple of changes to the operating system configuration that were performed during SAP installation on the primary node. In order to allow the system to run on other nodes, these changes need to be made to other cluster nodes, too. This section also includes steps to distribute the changes of the directory structure configuration performed in the section above.

NOTE: Repeat the cluster node synchronization steps for each node of the cluster that is different than the primary.

- Cluster Node Configuration - This section consists of steps performed on all the cluster nodes, regardless if the node is a primary node or a backup node.

NOTE: Repeat the cluster node configuration steps for each node of the cluster.

- External Application Server Host Configuration - This section consists of steps performed on any host outside of the cluster that runs another ABAP instance of the SAP system.

NOTE: Repeat this section for each host that has an external application server installed

Performing some of the iterations in parallel is fine. Use caution in any complex setup situation.

Rather than using the cut and paste mechanism you can also fill out the tables provided by first analyzing the primary host(s). Afterwards you can use the tables to synchronize all nodes.

Directory Structure Configuration

This section adds implementation practices to the architectural decisions made in the chapter "Storage Layout Considerations". If layout option 1 or option 2 is used, then the non-CFS directory structure conversion must be performed. An implementation based on HP LVM is described in this document. VxVM can similarly be used. Option 3 maps to the CFS configuration for SAP. In this case, usage of VxVM and CVM are mandatory.

Cluster Filesystem Configuration

Logon as root to the system where the SAP Central Instance is installed. If the database is installed on a different host, also open a shell as root on the database machine. Make sure that all SAP instances and the database got stopped manually.

NOTE: These steps are only required if HP Serviceguard Cluster File System is going to be used for the SAP instances.

Installation Step: IS009

Verify that the existing disk group layout is compliant with the needs of the Serviceguard package(s) as specified in the tables for option 3 of Chapter 2.

All directories that are labeled shared in these tables need to reside on logical volumes on a shared external storage system. The volume management needs to be VxVM based. The volumes can be converted using vxvmconvert(1m), if the SAP System got originally installed on HP LVM volumes instead.

All SAP Instances (CI, SCS, ASCS, D,...) must have instance numbers that are unique for the whole cluster. Exceptions are (A)REP instances that follow an old naming convention of sharing the instance ID with their corresponding (A)SCS instance. This naming convention should no longer be used, since it cannot be combined with the SAP startup framework. For a clustered instance with instance number `<INSTNR>` execute the command:

```
ls -d /usr/sap/???/*<INSTNR>
```

It should not reply any match on any node. Otherwise refer to the SAP documentation how to change instance IDs for the relevant instance type. It might be simpler to reinstall one of the conflicting instances.

Installation Step: IS030

Comment out the references to any file system that classified as a shared directory in Chapter 2 from the `/etc/fstab`.

Also make sure that there are no remaining entries for file systems converted in IS009.

MAXDB Database Step: SD040

This step can be skipped for MAXDB instances starting with versions 7.6.

MAXDB is not supported on CFS, but can be combined with SAP instances that use CFS. In this case, make sure you have mounted a sharable logical volume on `/sapdb/<DBSID>/wrk` as discussed in section MAXDB Storage Considerations in Chapter 2.

NOTE: `dbmcli` commands only work for users that have correct XUSER settings in place.

Change the path of the runtime directory of the MAXDB and move the files to the new logical volume accordingly.

```
cd /sapdb/data/wrk/<DBSID>

find . -depth -print | cpio -pd /sapdb/<DBSID>/wrk

cd ..

rm -r /sapdb/data/wrk/<DBSID>

dbmcli -d <DBSID> -u control,control

...> db_cold
```

```
...> param_directget RUNDIRECTORY

OK

RUNDIRECTORY /sapdb/data/wrk/<DBSID>

---

...> param_directput RUNDIRECTORY /sapdb/<DBSID>/wrk


OK
---

...>
```

Installation Step: IS049

If it does not yet exist, create a CVM/CFS system multi-node package and start it.

Integrate all SAP related CFS disk groups and file systems. The package dependencies to the system multi-node package get created automatically. The mount points need to be created manually on all alternate nodes first.

Here is an example for a disk group that was created with the following command sequence:

```
vxdg -s init dgC11 /dev/rdsk/c4t0d4
vxdg -g dgC11 set activation=sw
vxassist -g dgC11 make /dev/vx/rdsk/dgC11/cfs01 newfs -F vxfs \
    /dev/vx/rdsk/dgC11/cfs01
```

In order to put it into the Serviceguard cluster, the following command sequence is needed:

```
cfscluster config
cfscluster start
cfsdgadm add dgC11 all=sw
cfsmntadm add dgC11 cfs01 /usr/sap/<SID> all=rw
```

After these conversions, it should be possible to start the SAP System manually. If the installation was done using virtual IPs, then the additional packages defined in that step need to be started after starting the CFS packages, but prior to the SAP manual start.

Non-CFS Directory Structure Conversion

The main purpose of this section is to ensure the proper LVM layout and the right distribution of the different file systems that reside on shared disks. This section does not need to be consulted when using the HP Serviceguard Storage Management Suite with CFS and shared access Option 3.

Logon as root to the system where the SAP Central Instance is installed (primary host). If the database is installed on a different host, also open a shell as root on the database machine. Stop the SAP Application Server and the database if they are not already.

Installation Step: IS010

The device minor numbers of the shared volume groups must be the same on all cluster hosts.

They must differ from all other volume group device minor numbers used on the cluster hosts.

To keep track of that, record all the minor numbers already in use on the cluster hosts; see Table 3-1 - Hosts and Device Minor Numbers.

To find these minor device numbers on each cluster hosts, type:

```
ll /dev/vg*/group|cut -c 44-45
```

Table 3-1 Hosts and Device Minor Numbers

hostname	device minor numbers

Installation Step: IS020

Verify that the existing volume group layout is compliant with the needs of the Serviceguard package(s) as specified in the tables of Chapter 2.

1. Make sure that database specific file systems and Central Instance and/or System Central Services specific file systems are separated onto different volume groups. All directories that are labeled shared in these tables need to reside on logical volumes on a shared external storage system. The recommended separation in different volume groups can also be taken from these tables.

A default mount point for SAP applications is `/usr/sap/<SID>`. For Serviceguard, change this to `/usr/sap/<SID>/<INSTNAME><INR>`. If `/usr/sap/<SID>` is the mount point you have to move some files to a local logical volume and change the mount point.

For example for a SAP Central Instance:

```
mkdir /usr/sap/<SID>.new
cd /usr/sap/<SID>
bdf . # Remember the filesystem column.
# It will be referred to as <dev_path> later.
find . -depth -print|cpio -pd /usr/sap/<SID>.new
cd /
umount /usr/sap/<SID>
rmdir /usr/sap/<SID>
mv /usr/sap/<SID>.new /usr/sap/<SID>
chmod 751 /usr/sap/<SID>
chown <SID>adm:sapsys /usr/sap/<SID>
cd /usr/sap/<SID>/DVEBMGS<INSTNR>
rm -r * # be careful with this
cd ..
mount <dev_path> /usr/sap/<SID>/DVEBMGS<INSTNR>
cd DVEBMGS<INSTNR>
# remove everything that is different from DVEBMGS<INSTNR>,
```

```
ls
# Example: rm -r SYS
# rm -r D00
cd DVEBMGS<INSTNR>
find . -depth -print|cpio -pd /usr/sap/<SID>/DVEBMGS<INSTNR>
rm -r * # be careful with this
cd ..
rmdir DVEBMGS<INSTNR>
```

2. Mark all shared volume groups as members of the cluster. This only works if the cluster services are already available. For example:

```
cd /
# umount all logical volumes of the volume group
vgchange -a n <vg_name>
vgchange -c y <vg_name>
vgchange -a e <vg_name>
# remount the logical volumes
```

3. The device minor numbers must be different from all device minor numbers gathered on the other hosts. Verify this by comparing numbers listed in Table 1-3 Hosts and Device Minor Numbers to the numbers listed in the tables contained in chapter 2.

Create map files with the `vgexport` command and distribute them to all nodes in the cluster for all shared volume groups.

Installation Step: IS030

Comment out the references to any file system that classified as a shared directory in Chapter 2 from the *`/etc/fstab`****.***

MAXDB Database Step: SD040

Make sure you have mounted a sharable logical volume on *`/sapdb/<DBSID>/wrk`* ***as discussed in section MAXDB Storage Considerations in Chapter 2.***

Make sure you have mounted a sharable logical volume on `/sapdb/<DBSID>/wrk` as discussed in section MAXDB Storage Considerations in Chapter 2.

NOTE: This step can be skipped for MAXDB instances starting with versions 7.6.

Change the path of the runtime directory of the SAPDB and move the files to the new logical volume accordingly.

```
cd /sapdb/data/wrk/<DBSID>
find . -depth -print | cpio -pd /sapdb/<DBSID>/wrk
cd ..
rm -r/sapdb/data/wrk/<DBSID>
dbmcli -d <DBSID> -u control,control
...> param_directget RUNDIRECTORY
OK
RUNDIRECTORY /sapdb/data/wrk/<DBSID>
---
...> param_directput RUNDIRECTORY /sapdb/<DBSID>/wrk
OK
---
...>
```

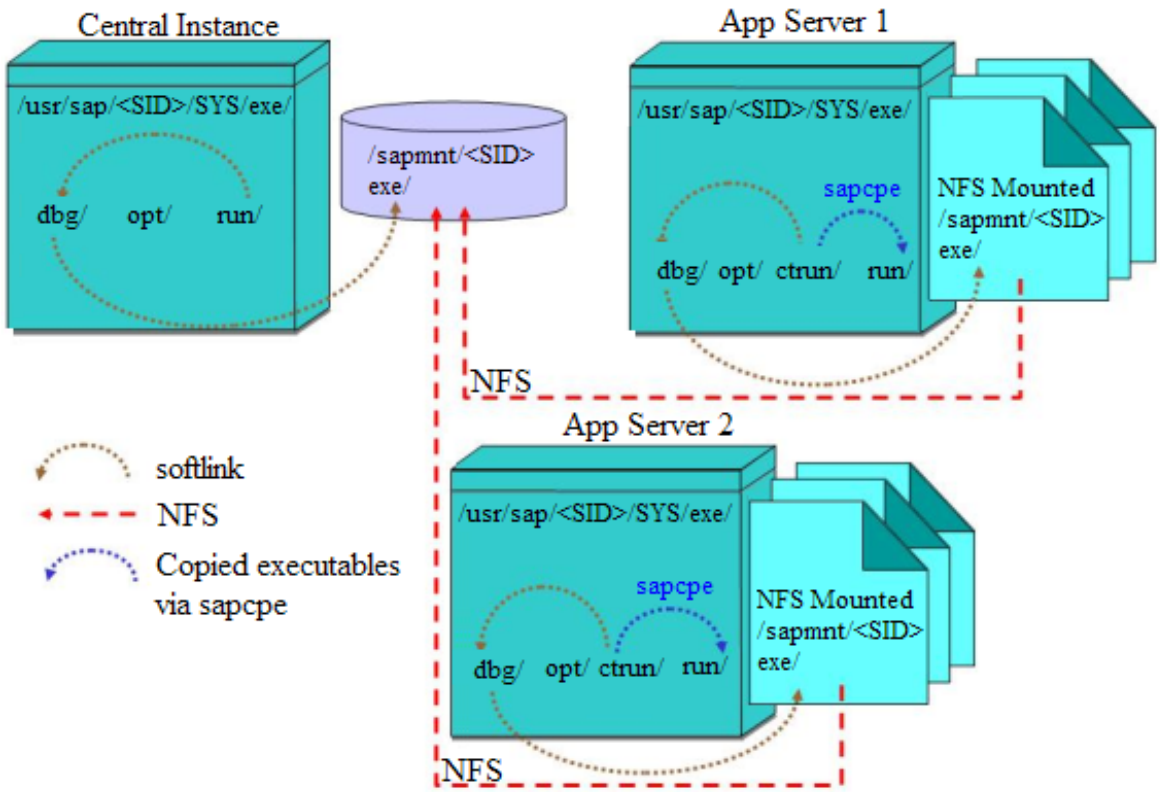
Installation Step: IS050

This step describes how to create a local copy of the SAP binaries. The step is mandatory for legacy packages. It is also required to do this, if a module-based package is used for a system with SAP kernel <7.0.

Check if the Central Instance host and all application servers have a directory named `/usr/sap/<SID>/SYS/exe/ctrun`. If the directory exists, this step can be skipped. The system is already using local executables through `sapcpe`.

To automatically synchronize local copies of the executables, SAP components deliver the sapcpe mechanism. With every startup of the instance, sapcpe matches new executables stored centrally with those stored locally. Figure 3-1 shows a SAP filesystem layout with sapcpe activated for two Application Servers.

Figure 3-1 sapcpe Mechanism for Executables



To create local executables, the SAP filesystem layout needs to be changed. The original link `/usr/sap/<SID>/SYS/exe/run` needs to be renamed to `/usr/sap/<SID>/SYS/exe/ctrun`. A new local directory `/usr/sap/<SID>/SYS/exe/run` will then be required to store the local copy. It needs to be initialized by copying the files `sapstart` and `saposcol` from the central executable directory `/sapmnt/<SID>/exe`. Make sure to match owner, group and permission settings.

Do not forget to set the s-bit of `saposcol`. To complete the process, a plain text configuration file is needed that tells sapcpe which files need to be included in the copying process.

A file of this type is usually called `sapcpeft`. If `sapcpeft` exists, sapcpe will automatically be executed as the first step of the next instance restart. It needs to be created in the central executable directory. For each additional file in the central executable directory it needs to contain a line with the following format:

```
<filename> local_copy check_exist
```

SAP also delivers template file lists within the central executable directory. These files have a filename of the format `*.lst`. The following files override `sapcpeft: frontend.lst, instance.lst, instancedb.lst, tools.lst and inhouse.lst`. Either the `*.lst` files should get removed, or they should be used instead of `sapcpeft`. In this case, they need to be changed to include all executables and libraries.

If the local executable directory only holds links, `sapcpe` is not configured correctly. It is not an option to manually copy the local executables in this case. The next instance restart would replace the local copies with symbolic links.

For latest information on how to utilize `sapcpe` refer to the SAP online documentation.

Installation Step: IS060

Clustered SAP Instances must have instance numbers that are unique for the whole cluster.

For a clustered instance with instance number `<INSTNR>` execute the command:

```
ls -d /usr/sap/???/*<INSTNR>
```

It should not reply any match on all nodes. Otherwise refer to the SAP documentation how to change instance IDs for the relevant instance type. It might be simpler to reinstall on of the conflicting instances.

Installation Step: IS065

If local executables are created (refer to step IS050) and failover nodes have no internal application server with local executables installed, distribute the directory tree *`/usr/sap/<SID>/SYS`* ***from the primary node.***

Do not use rcp(1), it will follow all links and copy a lot of files from the shared disks that are not needed.

For example, on the primary node:

```
cd /usr/sap/<SID>/SYS
find . -depth -print | cpio -o >/tmp/SYS.cpio
```

Use ftp(1) to copy the file over to the secondary node. On each alternate node:

```
su - <sid>adm
mkdir -p /usr/sap/<SID>/SYS
cd /usr/sap/<SID>/SYS
cpio -id </tmp/SYS.cpioexit
```

Cluster Node Synchronization

Repeat the steps in this section for each node of the cluster that is different than the primary.

Logon as root to the system where the SAP Central Instance is installed (primary host) and prepare a logon for each of its backup hosts - if not already available.

Installation Step: IS070

Open the groupfile, *`/etc/group`*, ***on the primary side.***

If any of the groups listed in Table 3-2 - Groupfile File Groups exist on the primary node and they do not exist on the backup node, copy them from the primary node to the backup node. If any group exists, verify that it has the same GID on both the primary and backup nodes. Merge the group members lists.

Table 3-2 Groupfile File Groups

Groups	GID	Group members
`sapsys`		
`sapinst`		
`sdba`		
`oper`		

Installation Step: IS080

Open the password file, *`/etc/passwd`*, ***on the primary side.***

If any of the users listed in Table 3-3 Password File Users exist on the primary node, recreate them on the backup node. Assign the users on the backup nodes the same user and group ID as the primary nodes.

NOTE: Beware of copying over into `/etc/passwd` if your HP-UX is running in Trusted System mode.

Table 3-3 Password File Users

username	UID	GID	home directory	shell
<sid>adm				
ora<SID>				
sqd<SID>				
sqa				
sap<dbsid>db				

Installation Step: IS090

Look at the service file, /etc/services, on the primary side.

Replicate all services listed in Table 3-4 Services on the Primary Node that exist on the primary node onto the backup node.

Table 3-4 Services on the Primary Node

Service name	Service port
sapdp<INR>	
sapdp<INR>s	
sapgw<INR>	
sapgw<INR>s	
sapms<SID>	
orasrv	
sapdb<INR>	
saphostctrl	
saphostctrls	

Installation Step: IS100

Change the HP-UX kernel on the backup node to meet the SAP requirements.

Compare the Tunable Parameters section of `/stand/system` on all nodes. All values on the backup nodes must reach or exceed the values of the primary node.

Install all HP-UX patches that are recommended for Serviceguard and patches recommended for SAP.

Build a new kernel with `mk_kernel(1m)` on each backup node if `/stand/system` was changed.

Installation Step: IS110

Copy the <sid>adm home directory to the backup node(s).

This is a local directory on each node. Default home directory path is `/home/<sid>adm`.

Installation Step: IS120

On the second node, in the <sid>adm home directory the start, stop and environment scripts need to be renamed.

If some of the scripts listed in the following do not exist, these steps can be skipped.

```
su - <sid>adm
mv startsap_<primary>_<INR> startsap_<secondary>_<INR>
mv stopsap_<primary>_<INR> stopsap_<secondary>_<INR>
mv .sapenv_<primary>.csh .sapenv_<secondary>.csh
mv .sapenv_<primary>.sh .sapenv_<secondary>.sh
mv .dbenv_<primary>.csh .dbenv_<secondary>.csh
```

```
mv .dbenv_<primary>.sh .dbenv_<secondary>.sh
mv .sapsrc_<primary>.csh .sapsrc_<secondary>.csh
mv .sapsrc_<primary>.sh .sapsrc_<secondary>.sh
mv .dbsrc_<primary>.csh .dbsrc_<secondary>.csh
mv .dbsrc_<primary>.sh .dbsrc_<secondary>.sh
```

The following statement should automate this activity for standard directory contents. Do not use a line break within the `awk` statement:

```
su - <sid>adm
ls -a|awk '/<primary>/ { system( sprintf( "mv %s %s\n", $0,\
gensub("<primary>", "<secondary>", 1 ))) }'
exit
```

Never use the relocatable address in these filenames. If an application server was already installed, do not overwrite any files which will start the application server. If the rc-files have been modified, correct any hard coded references to the primary hostname.

Installation Step: IS130

If the system has a SAP kernel release < 6.0:

On each host the files `/home/<sid>adm/startsap <local>_<INR>` and `/home/<sid>adm/stopsap_<local>_<INR>` exist and contain a line that specifies the start profile. After a standard installation of a Central Instance this line is similar to:

```
START_PROFILE="START_<INSTNAME><INR>_<primary>"
```

As`<sid>adm`, change the line individually on each node

On the primary host keep:

```
START_PROFILE="START_DVEBMGS<INR>_<primary>"
```

On the secondary (and any other potential failover) host change the value in both files to:

```
START_PROFILE="START_DVEBMGS<INR>_<secondary>"
```

Oracle Database Step: OR150

If the primary node has the ORACLE database installed:

Create additional links in `/oracle/<SID>` on the primary node. For example:

```
su - ora<sid>
ln .dbenv_<primary>.csh .dbenv_<secondary>.csh
ln .dbenv_<primary>.sh .dbenv_<secondary>.sh
exit
```

NOTE: If you are implementing an Application Server package make sure that you install the Oracle Client libraries locally on all nodes you run the package on. Refer to OSS Note 180430 for more details.

Oracle Database Step: OR160

If you are using ORACLE:

Create a mount point for the Oracle files on the alternate nodes if it is not already there. For example:

```
su - ora<sid>
mkdir -p /oracle/<SID>
exit
```

MAXDB Database Step: SD170

Create a mount point for the SAPDB files on the alternate nodes if it is not already there.

For example:

```
su - sqd<dbsid>
mkdir -p /sapdb/<DBSID>
```

Installation Step: IS190

In full CFS environments, this step can be omitted.

If CFS is only used for SAP and not for the database, then the volume group(s) that are required by the database need to be distributed to all cluster nodes.

Import the shared volume groups using the minor numbers specified in Table 1 - Instance Specific Volume Groups contained in chapter "Planning the Storage Layout".

The whole volume group distribution should be done using the command line interface. Do not use SAM. SAM will not create minor numbers in the intended fashion. Specify the device minor numbers explicitly by creating the group file manually.

On possible failover nodes, for example:

```
mkdir /dev/vgdb<SID>mknod /dev/vgdb<SID>/group c 64 0x080000
```

Now you can use `vgimport(1m)` with the map file created on the primary host during Step IS020.

Installation Step: IS210

Create a local directory for the `saposcol` temporary data on possible the backup nodes.

For example:

```
mkdir -p /usr/sap/tmp
```

Installation Step: IS220

Create a local mount point for each file system that was specified in chapter "Planning the Storage Layout to have shared disk" or HA NFS" access point

Refer to the tables in chapter 2 for Instance, System and Environment specific volume groups and the corresponding tables that represent entries for the required database type. Depending on the past usage of the hosts, some of the directories might already exist.

For non-CFS options only:

```
su - <sid>admmkdir -p /sapmnt/<SID>mkdir -p /usr/sap/transmkdir -p
/usr/sap/<SID>/<INSTNAME><INSTNR>exit
```

MAXDB Database Step: SD230

SAPDB database specific:

```
su - sqd<SID>
mkdir -p /sapdb/programs
mkdir -p /sapdb/data
mkdir -p /usr/spool/sql
exit
```

Ownership and permissions of these directories should be chosen equally to the already existing directories on the primary host.

MAXDB Database Step: SD235

For releases starting with MAXDB 7.5: Copy file `/etc/opt/sdb` to the alternate cluster nodes.

This file contains global path names for the MAXDB instance.

MAXDB Database Step: SD236

For releases starting with MAXDB 7.5:

Verify that the symbolic links listed below in directory`/var/spool/sql` exist on all cluster nodes running the database.

Cluster Node Configuration

Repeat the steps in this section for each node of the cluster.

Logon as root.

SGeSAP needs remote login to be enabled on all cluster hosts. The traditional way to achieve this is via remote shell commands. If security concerns prohibit this, it is also possible to use secure shell access instead.

Installation Step: IS240

Make sure that the required software packages are installed on all cluster nodes:

▲ Serviceguard Extension for SAP, T2803BA

The `swlist` command may be utilized to list available software on a cluster node

If a software component is missing install the required product depot files using the `swinstall` tool.

Installation Step: IS260

You need to allow remote access between cluster hosts.

This can be done by using remote shell `remsh(1)` or secure shell `ssh(1)` mechanisms.

If you allow remote access using the remote shell mechanism:

Create an `.rhosts` file in the home directories of the HP-UX users `root` and `<sid>adm`. Allow login for root as root from all nodes including the node you are logged into. Allow login for `root` and `<sid>adm` as `<sid>adm` from all nodes including the node you are logged into. Be careful with this step, many problems result from an incorrect setup of remote access.

Check the setup with `remsh` commands. If you have to provide a password, the `.rhosts` does not work.

Installation Step: IS270

If you allow remote access using the secure shell mechanism:

1. Check with swlist to see if `ssh` (T1471AA) is already installed on the system:

```
swlist | grep ssh
```

If not, it can be obtained from `http://www.software.hp.com/ISS_products_list.html`.

2. Create a public and private key for the root user:

```
ssh-keygen -t dsa
```

Executing this command creates a `.ssh` directory in the root user's home directory including the following files:

```
id_dsa
```

```
id_dsa.pub
```

The file `id_dsa.pub` contains the security information (public key) for the `user@host pair e.g. root@<local>`. This information needs to be added to the file `$HOME/.ssh/authorized_keys2` of the root and `<sid>adm user`.

Create these files if they are not already there. This will allow the root user on `<local>`to remotely execute commands via `ssh` under his own identity and under the identity of `<sid>adm` on all other relevant nodes.

On each cluster node where a SGeSAP package can run, test the remote access to all relevant systems as user root with the following commands:

```
ssh <hostN> date
```

```
ssh -l <sid>adm <hostN> date
```

Do these tests twice since the first `ssh` command between two user/host pairs usually requires a keyboard response to acknowledge the exchange of system level id keys.

Make sure that `$HOME/.ssh/authorized_keys2` is not writable by group and others. The same is valid for the complete path.

Permissions on ~<user> should be 755. Permissions on `~<user>/.ssh/authorized_keys2` must be 600 or 644.

Allowing group/other write access to `.ssh` or `authorized_keys2` will disable automatic authentication.

After successful installation, configuration and test of the secure shell communication `ssh` can be used by SGeSAP. This is done via setting the parameter `REM_COMM` to `ssh` in the SAP specific configuration file `sap.config` of section Configuration of the optional Application Server Handling.

```
# REM_COMM=ssh
```

```
# REM_COMM=remsh
```

Installation Step: IS280

If storage layout option 1 is used, create all SAP instance directories below `/export` ***as specified in Chapter 2.***

For example:

```
su - <sid>adm
mkdir -p /export/sapmnt/<SID>
mkdir -p /export/usr/sap/trans
exit
```

MAXDB Database Step: SD290

Create all MAXDB directories below `/export` ***as specified in Chapter 2.***

For example:

```
su - sqd<sid>
mkdir -p /export/sapdb/programs
mkdir -p /export/sapdb/data
mkdir -p /export/var/spool/sql/ini
exit
```

Installation Step: IS300

Each package needs at least one relocatable IP address to be specified.

In case there is a distinction between front-end, server, and LANs. It is likely that at least two addresses are needed per package. Dedicated heartbeat LANs would require additional addresses. Add all relocatable IP address information to `/etc/hosts`. Use Table 3-5 - Relocatable IP Address Information to record the addresses.

Table 3-5 Relocatable IP Address Information

name/aliases	IP address

Installation Step: IS310

Configure `/etc/nsswitch.conf` ***to avoid problems.***

If you use DNS:

For example:

```
hosts: files[NOTFOUND=continue UNAVAIL=continue TRYAGAIN=continue]dns
```

Installation Step: IS330

In some older SAP releases, during installation SAP appends some entries to the standard `.profile` ***files in the user home directories instead of using a new file defined by the SAP System. On HP-UX, by default, there is the following in the given profiles:***

```
set -u
```

This confuses the `.dbenv*.sh` and `.sapenv*.sh` files of the SAP System. They fail during execution if the environment is not setup properly. Using SGeSAP the package startup fails trying to bring up the database.

Search the `.profile of <sid>adm` and remove the `set -u`, if found.

External Application Server Host Configuration

Repeat the steps in this section for each host that has an external application server installed.

Logon as root.

Installation Step: IS340

If you want to use a secure connection with SSL, refer to IS270 on how to set up secure shell for an application server hosts.

Refer to IS260 if `remsh` is used instead.

Installation Step: IS350

Search `.profile` ***in the home directory of*** `<sid>adm` ***and remove the*** `set -u`***, if found.***

Installation Step: IS360

Add all relocatable IP address information to `/etc/hosts` ***on all nodes in the cluster.***

Modular Package Configuration

This section describes the cluster software configuration. Refer to the standard Serviceguard manual "Managing Serviceguard" to learn about creating and editing a cluster configuration file and how to apply it to initialize a cluster with `cmquerycl(1m)` and `cmapplyconf(1m)`. The following section assumes that a cluster is up and running on the intended primary node and all failover nodes.

Logon as root on any cluster host.

Installation Step: MS400

Create the SGeSAP package configuration file. The file might already exist, if the SAP System got installed onto virtual IP addresses.

The SGeSAP configuration file is usually created with one of the following commands:

`cmmakepkg -m sgesap/all > ./sap.config` — for a configuration file that covers all available SGeSAP module functionality and combines all SGeSAP modules in a single package.

`cmmakepkg -m sgesap/sapinstance > ./sap.config` — for a configuration file that is meant to create a package that clusters one or more SAP instances of a single SAP system.

`cmmakepkg -m sgesap/dbinstance > ./sap.config` — for a configuration file that is meant to create a package that clusters the database service underlying a single SAP system.

`cmmakepkg -m sgesap/sapinstance –m sgesap/dbinstance > ./sap.config` — for a configuration file that is meant to create a package that clusters a central system consisting of the database and one or more SAP instances of a single SAP system. All parts run on the same node at the same time.

NOTE: The System Central Service instances can be referred to by using SAP instance type naming conventions.

`cmmakepkg -m sgesap/scs >./scs.config` and `cmmakepkg -m sgesap/ers >./ers.config` — separates System Central Service and Enqueue Replication into two packages.

`cmmakepkg -m sgesap/scs -m sgesap/ers > ./sap.config` — would immediately issue an error message, because System Central Services and Enqueue Replication cannot share the same package.

NOTE: SGeSAP modules can be referred to by using SGeSAP legacy package types.

`cmmakepkg -m sgesap/db -m sgesap/ci > ./sap.config` — creates a single package configuration file for database and SAP instance(s) (one package concept).

`cmmakepkg -m sgesap/db > ./db.config` and `cmmakepkg -m sgesap/ci > ./sap.config` — separates database and SAP instances into two package configuration files (two package concept).

Optional Step: OS610

SGeSAP performs activities specific to the database you use. Specify the underlying database vendor using the DB parameter.

Possible options are: ORACLE and MAXDB. Refer to the SGeSAP release notes to see whether the required vendor is available for the SAP application component.

Example:

```
DB=ORACLE
```

Specify the relocatible IP address of the database instance. Be sure to use exactly the same syntax as configured in the IP[]-array in the package control file.

Example:

```
DBRELOC=0.0.0.0
```

In the subsection for the DB component there is an optional paragraph for Oracle and SAPDB/MAXDB database parameters. Depending on your need for special HA setups and configurations have a look at those parameters and their description.

Preparation Step: MS410

The generic Serviceguard parameters of the configuration file need to be edited.

Refer to the Managing Serviceguard User's Guide for general information about the generic file content: At least you'll want to specify `package_name`, `node_name`, `ip_address` and `monitored_subnet` in the file that you created with step MS400.

CFS based file systems are made available through the corresponding multinode packages as defined in Chapter 2 - Planning the Storage Layout. Package dependencies on the multinode CFS mountpoint packages should be defined to make sure the SAP file system(s) are available. For example:

```
dependency_name SG-CFS-MP-SAPEXE
dependency_condition SG-CFS-MP-SAPEXE=up
dependency_location same_node
```

For non-CFS shares, the volume groups as defined in Chapter 2 – Planning the Storage Layout might be directly connected to the SAP packages. That is usually the case for the instance and database homedirectories. Example:

```
vg vgC11
fs_name /dev/vgC11/lvol1
fs_directory /oracle/C11
:
fs_name /dev/vgC11/lovl2
fs_directory /usr/sap/C11/SCS01
:
```

Installation Step: MS420

The SGeSAP parameters of the configuration file need to be edited that influence the behavior for all instances that will be clustered by the package.

Only one mandatory parameter has to be specified: the SAP System ID `sap_system`. Each SGeSAP package can only failover instances that belongs to one and the same SAP System. Thus, the parameter needs to be specified exactly once. Example:

```
sap_system C11
```

Optional Step: OS421

The `rem_comm` and `priv_comm` values define the method to be used to remotely execute commands for SAP Application Server handling.

Setting the parameter `rem_comm` is optional. It can be set to `ssh` to rely on secure encrypted communications between untrusted hosts over an insecure network. Information on how to set up `ssh` for each node refer to Section Cluster Node Configuration. Default value of `rem_comm` is `remsh`.

The parameter `priv_comm` is a placeholder for future use. It will become effective if SAP clusters get supported that allow cross-subnet setups.

Optional Step: OS422

It is possible to influence the system cleanup behavior of SGeSAP.

Prior to any instance startup attempts the SGeSAP tries to free up unused or unimportant resources to make the startup more likely to succeed. A database package only frees up database related resources, a SAP Instance package only removes IPCs belonging to SAP administrators.

The following list summarizes how the behavior of SGeSAP is affected with different settings of the `cleanup_policy` parameter:

- `lazy` - no action, no cleanup of resources
- `normal` - removes orphaned resources as reported by SAP tools for the SAP system that is specified in `sap_system`. An obsolete ORACLE SGA is also removed if a database crash occurred.
- `strict` - uses HP-UX commands to free up system resources that belong to any SAP Instance of any SAP system on the host if the Instance is to be started soon.

NOTE: Do not use the strict policy unless it is required. Be aware that the strict option can crash running instances of different SAP systems on the backup host. Use this value only if you have a productive system that is much more important than any other SAP system you have. In this case a switchover of the productive system is more robust, but additional SAP systems will crash.

You can also use strict policy, if your SAP system is the only one running at the site and you are low on memory. Strict policy frees up more of its own shared memory segments than the normal policy does.

Optional Step: OS423

It is possible to influence the system cleanup behavior of SGeSAP.

Many SGeSAP package activities depend on system resources that are provided via mechanisms that are not directly handled by the package. If these resources are not available, a package operation could fail. Sometimes the resources are just temporarily unavailable and the package activity would succeed if delayed long enough. To allow that kind of synchronization, SGeSAP enters loops that poll missing resources regularly and delays or retries activities that depend on these resources. The package activity continues after the resource becomes available again or fails after a maximum of `retry_count` attempts to successfully finish the activity.

The default for the parameter is set to 5. It can be raised on demand, if the package logs indicate racing conditions with timing issues.

Subsection for the Oracle database component: OR425

Parameters that can be set to handle an Oracle single-instance database as part of a package with the `sgesap/dbinstance` or `sgesap/oracledb` module.

The package parameter `db_vendor` defines the underlying RDBMS database technology that is to be used with the SAP application. It is preset to oracle in `sgesap/oracledb`, but should otherwise be manually set to oracle. It is still optional to specify this parameter, but either `db_vendor` or `db_system` needs to be set in order to include the database in the failover package.

`db_system` determines the name of the database (schema) for SAP. Usually it is a three letter name, similar to a `sap_system` value (SID). If the value is not specified, but `db_vendor` has been set, a database with default `db_system=sap_system` is assumed (SAP's installation default), if `sap_system` is specified elsewhere in the package configuration.

Optionally, `listener_name` can be set if the Oracle listener is defined on a name different from the default value `LISTENER`.

Optionally, `listener_password` can be set if a password for the Oracle listener process is configured. The root user and any user with a defined Serviceguard access role (full admin, package admin or monitor) will be able to read this value.

`sgesap/dbinstance` can be used for databases of SAP JAVA-only, ABAP-only or dual stack instances. The module will detect all available SAP tools to handle the database and use them if possible. If no SAP tools are there or the SAP tools fail to work, a fallback is implemented to handle the database with generic tools of the database vendor.

Subsection for the MAXDB database component: SD426

Parameters that can be set to handle a MAXDB database as part of a package with the `sgesap/dbinstance` or `sgesap/maxdb` module.

The package parameter `db_vendor` defines the underlying RDBMS database technology that is to be used with the SAP application. It is preset to `maxdb` in `sgesap/maxdb`, but should otherwise be manually set to `maxdb`. It is still optional to specify this parameter, but either `db_vendor` or `db_system` needs to be set in order to include the database in the failover package.

db_system determines the name of the database (schema) for SAP. Usually it is a three letter name, similar to a sap_system value (SID). If the value is not specified, but db_vendor has been set, a database with default db_system=sap_system is assumed (SAP's installation default), if sap_system is specified elsewhere in the package configuration.

Optionally, maxdb_userkey sets the MAXDB userkey that is mapped for the operating system level administrator to the database control user (via XUSER settings). The value needs to be set if the default control userkey 'c' is not available to be used.

Subsection for the SAP component: MS427

Parameters that can be set to handle one or more SAP instances as part of a package with module sgesap/sapinstance.

A sap_instance defines any SAP Netweaver Application Server instance based on the SAP ABAP technology-stack, the SAP JAVA technology-stack or a combination of both. All parts of the instance are packaged together as a single entity. This includes instance parts like icman, workprocesses, SAP JVMs or an IGS. System Central Service instances or the corresponding Enqueue Replication Service instances can also be specified. SAP instances are defined by specifying the instance name and instance number.

The sap_virtual_hostname parameter corresponds to the virtual hostname that got specified during SAP installation. The SAP virtual hostname is a string value. It is not possible to specify the corresponding ipv4 or ipv6 address. If the string is empty, the DNS resolution of the first specified package ip_address parameter will be substituted. In this case, the script only works properly if reliable address resolution is available. Domain name extensions are not part of the virtual hostname.

SAP Enqueue Replication Service instances (ERS) have the purpose to keep track of the runtime status of a SAP Central Service instance. They allow for a more seamless failover experience. For each SAP Replication Service that is part of the package, the corresponding, replicated Central Service instance needs to be specified in the sap_replicated_instance parameter.

It is possible to put several SAP instances into one package by specifying sap_instance and the corresponding sap_virtual_hostname and sap_replicated_instance parameters more than once. SAP infrastructure software that does not come as part of an instance can not be specified here.

Example:

```
sap_instance D03
sap_instance J33
sap_instance DVEBMGS00
sap_virtual_hostname reloc2
sap_instance ERS02
sap_virtual_hostname reloc3
sap_replicated_instance SCS01
```

Optional Step: OS441

If the software health of a SAP System Central Service instance ([A]SCS) should be monitored in order to detect SAP software malfunctions with the cluster, the package configuration file needs to configure service monitoring for the SAP message server.

Monitoring Routines periodically check the availability and responsiveness of the SAP Message Server.

The message server should generally be configured to restart locally by specifying restart in the SAP (start) profile. If Serviceguard recognizes the SAP Message Server to be unavailable for a longer period of time, it assumes that the restart doesn't work or is accidentally not configured. Serviceguard will switch the package and try to restart on different hardware, usually the active enqueue replication server. SGeSAP service monitors will be paused within ten seconds if a debug file gets created for the package. For details see Chapter 6 –SGeSAP Cluster Administration. The debug file can be used to allow manual SAP instance shutdowns and startups. Make sure that all packaged SAP components are running when removing the debug file. Otherwise a failover will occur.

NOTE: The message server monitor works without command line parameters in the service command string `service_cmd`. It will configure itself automatically during startup and monitors all message servers that are clustered within the package.

Example entries for the package configuration file:

```
service_name scsC11ms
service_cmd "/opt/cmcluster/sap/SID/sapms.mon"
service_restart 2
service_fail_fast_enabled no
service_halt_timeout 300
```

Optional Step: OS446

If the enqueue service of the SAP System Central Services should also be handled, the automated Replicated Enqueue handling including follow-and-push failover semantics needs to be activated for a [A]SCS/ERS package pair:

The SGeSAP ENQueue Operation Resource (ENQOR) needs to be specified for each SCS, ASCS or ERS instance. The monitoring resource is provided by SGeSAP within the HP-UX Event Monitoring System (EMS). It does not need to be created. It is available, if the system has an SAP system installed, that uses [A]SCS/ERS instance pairs.

Available resources can be checked with the command:

```
resls /applications/sap/enqor
```

The required resource has the name `/applications/sap/enqor/<SID >ers<instnr>` for applications based on SAP kernel SAP 7.x.

NOTE: The required resource has the name `/applications/sap/enqor/<SID >scs` or `/applications/sap/enqor/<SID>ascs` for SAP applications based on kernel 4.x or 6.x.

Example entries for a package that clusters a [A]SCS instance that is replicated within instance ERS21:

```
resource_name /applications/sap/enqor/C11ers21
resource_polling_interval 30
resource_start automatic
resource_up_value != idle
```

NOTE: If the package clusters SCS and ASCS instance, two enqor resources need to be defined.

Example entries for a package that clusters the replication instance ERS21:

```
resource_name /applications/sap/enqor/C11ers21
resource_polling_interval 30
resource_start automatic
resource_up_value != serving
```

Subsection for external Application Server handling: OS471

Parameters that can be set to influence non-critical additional instances on cluster nodes or outside of the cluster ("external Application Servers") if module sgesap/sapextinstance is used.

In certain setups, it is necessary to free up resources on the failover node to allow the failover to succeed. Often, this includes the stop of less important SAP Systems, namely consolidation or development environments.

The following parameters may be used to configure special treatment of SAP Application Server instances during package start, halt or failover. These Application Servers need not necessarily be virtualized. They must not be clustered themselves. Clustered SAP Instances can be influenced by specifying Serviceguard package dependencies.

An attempt can be triggered to start, stop, restart or notify instances that belong to the same SAP system as the instances in the package. An attempt can be triggered to shutdown instances that belong to any other SAP system (details are given below).

All the triggered attempts are considered to be non-critical. If any triggered attempt fails, it doesn't cause failure of the ongoing package operation. Don't use these parameters for non-redundant resources and single points of failure.

A `sap_ext_instance` defines any SAP Netweaver Application Server instance based on the SAP ABAP technology-stack, the SAP JAVA technology-stack or a combination of both. All parts of the instance are treated together as a single entity. This includes instance parts like icman, workprocesses, SAP JVMs or an IGS. SAP infrastructure software that does not come as part of an instance can not be specified here.

`sap_ext_system` defines the unique SAP System Identifier (SAP SID) to which the corresponding `sap_ext_instance` is associated. If no value is specified, the setting of `sap_system` is assumed.

`sap_ext_host` defines the hostname on which the instance resides (which might be inside or outside of the cluster). It is also possible to specify a virtual hostname. The host can run a Linux or HP-UX operating environment. If the value is not specified, the local hostname computed during a package operation gets substituted.

`sap_ext_treat` defines the way the instance is treated if the status of the package changes. `sap_ext_treat` values are of a sequence of five 'y' and 'n' characters, each position representing the activation or deactiviation of a feature.

Set the 1. position to 'y' to automatically make an attempt to start the instance with each package startup operation.

Set the 2. position to 'y' to automatically make an attempt to stop the instance with each package shutdown operation.

Set the 3. position to 'y' to automatically make an attempt to restart the instance with each package failover operation.

Set the 4. position to 'y' to automatically shut down the instance if the package starts on the node on which it is running.

Position 5 is currently not in use.

If a specified instance is a Central Instance and other specified instances depend on it, the shutdown sequence will make sure that this Central Instance will always be started before and stopped after the depending instances. Apart from that rule, several instance operations will be triggered in parallel.

The first three positions in the treatment string are only effective if the instances do belong to the SAP system to which the package is associated `(sap_ext_system=sap_system)`. The fourth position can also be set for any other SAP system, in order to allow system consolidation.

Example 1:

The package is associated to SAP System SG1. The primary node is also running a non-clustered ABAP Dialog Instance with instance ID 01. It should be stopped and started with manual package operations. In case of a failover, a restart attempt should be made on the primary node (if the primary node is reachable from the secondary). There is a second instance D01 on a server outside of the cluster that should similarly be started, stopped and restarted.

```
sap_ext_instance
D01 sap_ext_host
node1 sap_ext_treat yyynn
sap_ext_instance D02
sap_ext_host hostname1
sap_ext_treat yyynn
```

Example 2:

The failover node is running a central, non-clustered testsystem QAS and a dialog instance D03 of the clustered SG1. All this should be stopped in case of a failover to the node, in order to free up resources.

```
sap_ext_instance DVEBMGS10
sap_ext_system QAS
```

```
sap_ext_host node2
sap_ext_treat nnnyn
sap_ext_instance D03
sap_ext_host node2
sap_ext_treat yyyyn
```

Subsection for additional SAP infrastructure software handling: OS472

Parameters to influence SAP software that runs outside of SAP Application Server Instances come with module `sgesap/sapinfra`.

`sap_infra_sw_type` defines a SAP infrastructure software component that is to be covered by the package. Such software pieces often support a specific SAP Netweaver Application Server, but they won't get started and/or stopped as part of the Instance. SAP Netweaver Application Server Instances are not specified here, but by using the `sap_instance` attribute. Legal values for `sap_infra_sw_type`:

`saposcol` = a SAP operating system monitor collector

`sapccmsr` = a SAP additional monitor collector

`rfcadapter` = a SAP XI/PI/EAI remote function call adapter

The following values can be specified more than once:

`saprouter` = a SAP software network routing tool

`biamaster` = a SAP BIA master nameserver (not for production use)

`sap_infra_treat` defines whether the tool will only be started/notified with the package startup, or whether it will also be stopped as part of a package shutdown (default). Possible values are `startonly` and `startnstop`.

`sap_infra_sw_params` defines additional command line parameters to be called with the tool.

A `sap_infra_sw_host` value can be added to specify the hostname where to start a BIA master instance. This parameter will be ignored for other infrastructure components, which always get started/stopped locally.

Examples:

```
sap_infra_sw_type saposcol
sap_infra_sw_type saprouter
sap_infra_sw_treat startnstop
sap_infra_sw_params "-H virtual_IP -W 20000\
-R /sapmnt/C11/profile/saprouttab\
-T /sapmnt/C11/profile/dev_rout1"
sap_infra_sw_type sapccmsr
sap_infra_sw_params /sapmnt/C11/profile/ccmsr_profilename
```

Installation Step: IS473

Integrate the SAP startup framework with the cluster.

SAP introduced a startup control framework with Netweaver 2004s and SAP Application Server 7.0 that enables the management of SAP instances from third party products using a defined application interface. This control framework is implemented using a daemon process that is either started by:

- the operating system during boot time or
- the SAP startup script when the instance is started.

During the installation of a SAP Netweaver 2004s Web AS, the SAPlinst installer edits a file in `/usr/sap/sapservices` and adds a line for each `sapstartservice` instance. During boot time an init script is executed located in:

```
/sbin/init.d/sapinit referenced by /sbin/rc3.d/S<###>sapinit
```

The `sapinit` script reads the content in file `/usr/sap/sapservices` and starts a `sapstartsrv` for each instance during OS boot time. This can lead to error output in the boot messages log file as file systems for exclusive shared storage is not available at that time.

If the use of the SAP control framework is not required, then remove the reference link from the `sapinit` script. Furthermore, any running `sapstartsrv` processes can be killed from the process list. For example:

```
rm /sbin/rc3.d/S<###>sapinit
ps -ef | grep sapstartsrv
kill <PID_of_sapstartsrv>
```

Module-based SGeSAP packages will handle the `sapstart` service agent automatically. For SAP releases based on kernel 7.10 or higher, SGeSAP can also register an instance with the startup framework as part of the package startup operation and de-register it as part of the package halt.

If the service agent is started via SGeSAP and configured to autostart the SAP instance, SGeSAP will monitor the operation to make sure that the instance startup succeeds.

Installation Step: IS474

The configuration file needs to be applied and the package started.

This step assumes that the cluster as such is already configured and started. Please refer to the Managing Serviceguard user's guide if more details are required.

The SGeSAP package configuration will be distributed to all nodes, verified on all nodes, and activated by issuing the following command on the package configuration file that was created and customized above:

```
cmapplyconf -P ./sap.config
```

Legacy Package Configuration

This section describes the cluster software configuration with the following topics:

- Serviceguard Configuration
- SGeSAP Configuration
- Global Default Settings

Serviceguard Configuration

Refer to the standard Serviceguard manual Managing Serviceguard to learn about creating and editing a cluster configuration file and how to apply it to initialize a cluster with `cmquerycl(1m)` and `cmapplyconf(1m)`. The following section assumes that a cluster is up and running on the intended primary node and all failover nodes.

Logon as root on the primary host.

Installation Step: LS400

Create a Serviceguard package directory. This directory will be used by all packages that belong to the SAP System with a specific SAP System *`ID <SID>`***.**

```
mkdir -p /etc/cmcluster/<SID>
```

Installation Step: LS410

The *`swinstall`* ***process copied relevant files to a staging area at*** *`/opt/cmcluster/sap`* ***for reference.***

The following files need to be copied from the repository to the SAP package directory:

```
cp /opt/cmcluster/sap/sap.functions/etc/cmcluster/sap.functions
cp /opt/cmcluster/sap/SID/sap.config/etc/cmcluster/<SID>/sap.config
cp /opt/cmcluster/sap/SID/customer.functions\
/etc/cmcluster/<SID>/customer.functions
```

SGeSAP automatically detects the package setup with the desired SAP component configured.

```
cp /opt/cmcluster/sap/SID/sapwas.cntl/etc/cmcluster/<SID>/sapwas.cntl
```

Installation Step: LS420

Enter the package directory *`/etc/cmcluster/<SID>`***. Create standard package configuration and control files for each package. The files might already exist, if the SAP System got installed onto virtual IP addresses.**

Decide which package types are needed in the high availability environment depending on the installed SAP System generate a control and configuration file for each package. These files should always be created using the standard Serviceguard commands. The package types are recommended to be named `<pkg_name>.control.script` for a control file and `<pkg_name>.config` for a configuration file.

For any kind of package (substitute `<pkg_type>` with your package type):

```
cmmakepkg -s/etc/cmcluster/<SID>/<pkg_name>.control.script
cmmakepkg -p/etc/cmcluster/<SID>/<pkg_name>.config
```

Installation Step: LS430

The created configuration file(s) need to be edited.

Refer to the Managing Serviceguard User's Guide for general information about the file content. The naming convention `<pkg_name>` should be used for the `PACKAGE_NAME` parameter.

For example:

```
PACKAGE NAME dbciC11
```

Specify NODE_NAME entries for all hosts on which the package should be able to run.

Specify the package control script names in the package configuration files:

Specify in `/etc/cmcluster/<SID>/<pkg_name>.config`:

```
RUN_SCRIPT /etc/cmcluster/<SID>/<pkg_name>.control.script
HALT_SCRIPT /etc/cmcluster/ <SID>/<pkg_name>.control.script
```

Specify subnets to be monitored in the SUBNET section.

Installation Step: OS435

This ensures a successful package start only if the required CFS file system(s) are available.

```
DEPENDENCY_NAME SG-CFS-MP-1
DEPENDENCY_CONDITION SG-CFS-MP-1=UP
DEPENDENCY_LOCATION SAME_NODE
DEPENDENCY_NAME SG-CFS-MP-2
DEPENDENCY_CONDITION SG-CFS-MP-2=UP
DEPENDENCY_LOCATION SAME_NODE
```

Optional Step: OS440

If the SAP message server or SAP dispatcher should be monitored as Serviceguard service, the *<pkg_name>.config* ***file needs to configure the monitoring.***

The recommended naming for the service is <pkg_name>ms or <pkg_name>disp.

Monitoring Routines periodically check the availability and responsiveness of the SAP Message Server or SAP dispatcher. If it recognizes the SAP Message Server or SAP dispatcher to be unavailable, Serviceguard will switch the package and try to restart on different hardware since the SAP Message Server or SAP dispatcher are crucial components of a SAP Central Instance.

First, the monitoring scripts need to be copied to the current package directory. For a SAP System C11:

```
cp /opt/cmcluster/sap/SID/sapms.mon /etc/cmcluster/C11
cp /opt/cmcluster/sap/SID/sapdisp.mon /etc/cmcluster/C11
```

The monitor will be paused within ten seconds if a debug file gets created for the package:

```
touch /etc/cmcluster/<SID>/debug
```

The debug file can be used to allow manual SAP instance shutdowns and startups.

NOTE: The manual startup using common startsap/stopsap scripts may not work correctly with Netweaver 2004s based Application Servers. In this case the instances need to started by directly using the sapstart binary passing the start profile as a parameter. For example:

```
sapstart pf=/sapmnt/<SID>/profile/START_\
<INAME><INR>_<RELOC_IP/hostname>
```

The service monitor will not be able to detect a stopsap command and will consider a manual instance shutdown to be a failure. The package will then fail over the instance to another node. The existence of the debug file can prevent that. If the debug file gets removed, the service monitor continues to run. Make sure that the packaged SAP instance is already running when removing the debug file or an immediate failover will occur. As long as the debug file exists locally, a package shutdown would not try to stop the SAP instance or its database. A local restart of the package would only take care of the HA NFS portion. The instance would not be started with the package and the monitor would be started in paused mode.

Example entries in <pkg_name>.config:

Message Server Monitor:

```
SERVICE_NAME ciC11ms
SERVICE_FAIL_FAST_ENABLED YES
SERVICE_HALT_TIMEOUT 300
```

Dispatcher Monitor:

```
SERVICE_NAME ciC11disp
SERVICE_FAIL_FAST_ENABLED YES
SERVICE_HALT_TIMEOUT 300
```

The service name and command referencing the SAP message server monitor and SAP dispatcher monitor must be specified in <pkg_name>.control.script in order to use the service defined in the config files

<pkg_name>.config. If no configuration parameters are provided with the service command, the monitors will default to the settings for a ABAP DVEBMGS Central Instance if possible.

Example entries in <pkg_name>.control.script:

```
SEVICE_NAME[0]="ciC11ms"
SERVICE_CMD[0]="/etc/cmcluster/C11/sapms.mon\<MS_SERVICEPORT>"
SEVICE_NAME[1]="ciC11disp"
SERVICE_CMD[1]="/etc/cmcluster/C11/sapdisp.mon <INSTNAME>\
<INSTNR> <hostname>"
```

Optional Step: OS445

In case you want to automate Replicated Enqueue handling and use follow-and-push failover semantics for[A]SCS/[A]REP packages:

For this to work with legacy packages, section 1 of the SGeSAP configuration file needs to reside in the file /etc/cmcluster/<SID>/sap.config This is the default approach even without the follow-and-push and it can always be done, also if several SGeSAP packages share this package directory. Other sections of the file can be moved to package specific files. /etc/cmcluster/<SID>/sap<packagename>.config files if required.

The SGeSAP ENQueue Operation Resource needs to be specified for each [A]SCS package and for each [A]REP replication package in the package configuration files.

Example entries for a ASCS package configuration file:

```
RESOURCE_NAME /applications/sap/enqor/C11ascs
RESOURCE_POLLING_INTERVAL 30
RESOURCE_START AUTOMATIC
RESOURCE_UP_VALUE != "IDLE"
```

Example entries for an AREP package configuration file:

```
RESOURCE_NAME /applications/sap/enqor/C11ascs
RESOURCE_POLLING_INTERVAL 60
RESOURCE_START AUTOMATIC
RESOURCE_UP_VALUE != "SERVING"
```

You will need to copy /opt/cmcluster/sap/sap.config to /etc/cmcluster if you use non-default package names. Default package names for setups with Replicated Enqueue are ascs<SID>, scs<SID> and ers<INR><SID>. Within the file, any non-default names have to be specified explicitly. If these values have to be changed during operation, the cmenqord process needs to be notified by sending it the signal USR1 via the kill command.

For SAP kernel 4.6 or 6.x and resources

```
/applications/sap/enqor/<SID>
/applications/sap/enqor/<SID>ascs
/applications/sap/enqor/<SID>scs
```

use:

```
ENQOR_ASCS_PKGNAME_<SID>=<non-default name_1>
ENQOR_SCS_PKGNAME_<SID>=<non-default name_2>
ENQOR_AREP_PKGNAME_<SID>=<non-default name_3>
ENQOR_REP_PKGNAME_<SID>=<non-default name_4>
```

Starting with SAP kernel 7.x and resources

```
/applications/sap/enqor/<SID>ers<instnr>
```

use:

```
ENQ_SCS_ERS<INSTNR>_PKGNAME_<SID>=<non-default name_2>
ENQ_ERS_ERS<INSTNR>_PKGNAME_<SID>=<non-default name_4>
```

<INSTNR> needs to be replaced by the SAP instance number.

<SID> needs to be replaced with the 3-letter SAP system identifier.

<non-default name_x> can be any valid Serviceguard package name.

Example:

```
ENQOR_SCS_PKGNAME_C11=foobar
ENQOR_REP_PKGNAME_C11=foorep
```

For SAP kernel 7.x; instances SCS00 and ERS01:

```
ENQOR_SCS_ERS01_PKGNAME_C11=foobar
ENQOR_ERS_ERS01_PKGNAME_C11=fooers
```

Optional Step: OS450

For non-CFS shares, it is recommended to set *AUTO_VG_ACTIVATE=0* ***in*** */etc/lvmrc****.***

Edit the `custom_vg_activation()` function if needed. Distribute the file to all cluster nodes.

Optional Step: OS460

It is recommended to set *AUTOSTART_CMCLD=1* ***in*** */etc/rc.config.d/cmcluster****.***

Distribute the file to all cluster nodes.

Installation Step: LS470

>Define volume groups, logical volumes, IP addresses and subnets in the package control scripts *<pkg_name>.control.script****.***

Fill the `IP[*]` and `SUBNET[*]` array with the IP addresses and subnet addresses that the package is attached as well.

When using the HP Serviceguard Storage Management Suite with CFS and shared access Option 3 (Chapter 2 - Planning the Storage Layout) the definition and configuration of logical volumes in the package control script might not be necessary for the SAP components. The CFS based file systems are made available through the corresponding multinode packages as defined in Chapter 2 - Planning the Storage Layout. For non-CFS shares, the volume groups are directly connected to the SAP packages. The file systems that are specified in the LV[*], FS[*] and FS_MOUNT_OPT[*] array in this case are not identical to the file systems that are for HA NFS usage. For example:

```
LV[0]="/dev/vgdbC11/lvoracle"; FS[0]="/oracle/C11"; FS_MOUNT_OPT[0]="-o rw"
LV[1]="/dev/vgdbC11/lvsapmnt"; FS[1]="/export/sapmnt/C11"; FS_MOUNT_OPT[1]="-o
rw"
LV[2]="/dev/vgC11/lvusrsap"; FS[2]="/usr/sap/C11/DVEBMGS20"; FS_MOUNT_OPT[2]="-o
rw"
LV[3]="/dev/vgsapnfs/lvtrans"; FS[3]="/export/usr/sap/trans";
FS_MOUNT_OPT[3]="-o rw"
```

Installation Step: LS480

To enable the SAP specific scripts change the *customer_defined_commands* ***sections of the package control script(s)*** *<pkg_name>.control.script****:***

```
function customer_defined_run_cmds
{
. /etc/cmcluster/<SID>/sapwas.cntl start <SID>
test_return 51
}
function customer_defined_halt_cmds
{
. /etc/cmcluster/<SID>/sapwas.cntl stop <SID>
test_return 52
}
```

The SAP specific control file sapwas.cntl needs to arguments, which are the MODE (start/stop) and the SAP System ID (SID). Don't omit the leading period sign in each line that calls sapwas.cntl.

Installation Step: LS500

Distribute the package setup to all failover nodes.

For example, you have to create package directories /etc/cmcluster<SID> on all backup nodes, copy all integration files below /etc/cmcluster/<SID> from the primary host's package directory to the backup host's package directory using rcp(1) or cmcp(1m) and similarly copy /etc/cmcluster/sap.functions from the primary host to the same location on the backup hosts.

Installation Step: LS510

Use cmapplyconf(1m) to add the newly configured package(s) to the cluster.

NOTE: If you plan to use a sapnfs package as central NFS service, specify this package in the last position of the cmapplyconf command. Later, if you force a shutdown of the whole cluster with cmhaltcl -f this package is the last one stopped. This prevents global directories from disappearing before all SAP components in the cluster have completed their shutdown.

Verify that the setup works correctly to this point. All operating system resources should be able to switch between the hosts. To do this, you can temporarily create debug files /etc/cmcluster/<SID>/debug. This prevents SAP specific package startup operations to be executed for each node on which this file gets created. SAP specific package operations would not succeed at this point of the integration. At this point of the integration, the package control log of the shutdown procedures will contain error messages, because the existence of a debug file does not prevent all of the instance shutdown attempts that come as part of the package operation. For example, database shutdowns are never skipped by the package logic. This is to prevent that careless testing causes unexpected database crashes. Such crashes might require undesirable database recovery efforts. Do not forget to remove all debug files after the tests.

SGeSAP Configuration

This section deals with the configuration of the SAP specifics of the Serviceguard packages. In Chapter 1 various sample scenarios and design considerations that apply to common setups were introduced. The mapping of this design to SGeSAP is done in a SAP specific configuration file called sap.config. The file is structured into four main sections that serve different purposes:

Section 1: Specification of the Packaged SAP Components

Section 2: Configuration of Application Server Handling

Section 3: Optional Parameters and Customizable Functions

Section 4: Global Defaults

Specification of the Packaged SAP Components

For each type of potentially mission-critical SAP software components there exists a set of configuration parameters in section 1 of the sap.config file. The information delivered here will be specified exactly once and it will configure all packaged components of a <SID> within the cluster. This is a central place of configuration, even if the intention is to divide the components into several different packages. The mapping of components to packages will be done automatically. There is one subsection per package type:

- db: a database utilized SAP ABAP or SAP add-in installations. Do not combine a database with a liveCache in a single package.
- ci: a SAP Instance that has SPOF services defined. This might be a Central Instance with integrated Enqueue and Message Service (DVEBMGS). It might also be a standalone Enqueue and Message Service as part of a separate ABAP System Central Service instance (ASCS). A Replicated Enqueue setup can be implemented by configuring at least one additional package of type RE in combination with this CI.
- arep: an ASCS Replicated Enqueue instance.
- rep: an SCS Replicated Enqueue instance.
- d: one or more virtualized SAP ABAP Application Instances that are considered mission-critical due to special use cases.

- jdb: a database exclusively utilized by J2EE engines that are not part of Application Server Instances. Do not combine a database with a liveCache in a single package.
- jci: a SAP JAVA System Central Services Instance that provides Enqueue and Message Service to SAP J2EE engines that might or might not be part of SAP Application Servers (SCS).
- jd: one or more virtualized SAP JAVA Application Instances.

NOTE: It is not allowed to specify a db and a jdb component as part of the same package. It is not allowed to specify a [j]ci and an [a]rep component as part of the same package. Except Dialog Instance components of type d, each component can be specified once at most. Apart from these exceptions, any subset of the SAP mission critical components can be maintained in sap.config to be part of one or more packages.

Logon to the primary host as root and open the file /etc/cmcluster/<SID>/sap.config with a text editor.

Installation Step: LS600

Specify the relocatible IP address where the HA NFS shared filesystems can be reached. Format is IPv4: nnn.nnn.nnn.nnn.

This value specifies the default for commonly shared filesystems. An override is possible for the SAPDB ini-directory, the global transport directory and the central executable directory individually by specifying values for SQLRELOC, TRANSRELOC, CTEXERELOC.

NOTE: All xxxRELOC parameters have to use the same syntax as the IP[]-array in the package control file.

When using the HP Serviceguard Storage Management Suite with CFS and shared access Option 3 (Chapter 2 - Planning the Storage Layout) the NFSRELOC does not need to be specified.

Example:

```
NFSRELOC=0.0.0.0
```

Subsection for the DB component: OS610

SGeSAP performs activities specific to the database you use. Specify the underlying database vendor using the DB parameter.

Possible options are: ORACLE and MAXDB. Refer to the SGeSAP release notes to see whether the required vendor is available for the SAP application component.

Example:

```
DB=ORACLE
```

Specify the relocatible IP address of the database instance. Be sure to use exactly the same syntax as configured in the IP[]-array in the package control file.

Example:

```
DBRELOC=0.0.0.0
```

In the subsection for the DB component there is an optional paragraph for Oracle and SAPDB/MAXDB database parameters. Depending on your need for special HA setups and configurations have a look at those parameters and their description.

Subsection for the CI component: OS620

Provide information about the Central Instance if it will be protected by SGeSAP. Set the parameter CINAME to the SAP instance name where ABAP Enqueue and Message Service are located.

This usually is the Central Instance or the ABAP System Central Service. The name is either DVEBMGS or ASCS. Note that the instance ID as defined by CINR is not part of CINAME.

Set the parameter CINR to the Instance ID of your Central Instance where ABAP Enqueue and Message Service are located. This is most likely a SAP Central Instance or a ABAP System Central Service Instance.

The relocatible IP address of the instance should be specified in CIRELOC.

Example:

```
CINAME=DVEBMGS
CINR=00
CIRELOC=0.0.0.0
```

In the subsection for the CI component there is a paragraph for optional parameters. Depending on your need for special HA setups and configurations have a look at those parameters and their description.

Subsection for the AREP component: OS630

SGeSAP supports SAP stand-alone Enqueue Service with Enqueue Replication.

It's important to distinguish between the two components: the stand-alone Enqueue Service and the replication. The stand-alone Enqueue Service is part of the ci or jci component. The arep component refers to the replication unit for protecting ABAP System Services.

Specify the instance name in AREPNAME, the instance ID number in AREPNR and the relocatible IP address of the SAP instance for the replication service in AREPRELOC.

Example:

```
AREPNAME=ERS
AREPNR=01
AREPRELOC=0.0.0.0
```

Subsection for the D component: OS640

A set of SAP ABAP Application Servers and ABAP Dialog Instance can be configured in each Serviceguard package.

Set the relocatible IP address of the dialog instance in DRELOC, the dialog instance name in DNAME and the instance ID number in DNR.

For example:

```
DRELOC[0]=0.0.0.0
DNAME[0]=D
DNR[0]=53
DRELOC[1]=0.0.0.0
DNAME[1]=D
DNR[1]=54
```

Subsection for the JDB component: OS650

The RDBMS for a J2EE Engine can be configured in this section.

Default SAP installations load the J2EE database scheme into the same database that the ABAP Engine is using. In this case, no JDB component needs to be specified and this configuration section does not need to be maintained.

For the parameter JDB, the underlying RDBMS is to be specified. Supported databases are ORACLE or SAPDB, for example:

```
JDB=ORACLE
```

Also, a relocatible IP address has to be configured for the database for J2EE applications in JDBRELOC.

Additionally for the J2EE RDBMS, the database instance ID name in JDBSID and database administrator in JDBADM is required to be specified:

```
JDBSID=EP6
JDBADM=oraep6
```

In the subsection for the JDB component there is a paragraph for optional parameters. Depending on your need for special HA setups and configurations have a look at those parameters and their description.

Subsection for the JCI component: OS660

SAP J2EE Engine Enqueue and Message Services are isolated in the System Central Service Instance.

SDM is not part of this instance. Specify the instance name in JCINAME, the instance ID number in JCINR and relocatible IP address in JCIRELOC, as in the following example:

JCINAME=SCS

JCINR=01

JCIRELOC=0.0.0.0

Subsection for the REP component: OS665

SGeSAP supports SAP stand-alone Enqueue Service with Enqueue Replication.

It's important to distinguish between the two components: the stand-alone Enqueue Service and the replication. The stand-alone Enqueue Service is part of the `ci` or `jci` component. The `rep` component refers to the replication unit for protecting JAVA System Central Services.Specify the instance name in REPNAME, the instance ID number in REPNR and the relocatible IP address of the SAP instance for the replication service in REPRELOC. For example:

REPNAME=ERS

REPNR=00

REPRELOC=0.0.0.0

Subsection for the JD component: OS666

A set of SAP JAVA Application Servers can be configured in each Serviceguard package.

Set the relocatible IP address of the dialog instance in JDRELOC, the dialog instance name in JDNAME and the instance ID number in JDNR. The startup of the J2EE engines get triggered with the package, but the successful run of the JAVA applications does not yet get verified.

For example:

JDRELOC[0]=0.0.0.0

JDNAME[0]=JC

JDNR[0]=10

JDRELOC[1]=0.0.0.0

JDNAME[1]=J

JDNR[1]=11

Configuration of Application Server Handling

In more complicated setups, there can be a `sap<pkg_name>.config` file for each package. For example a Dialog Instance package can have its own `sap<pkg_name>.config` configured to start additional non-critical Dialog Instances, whereas this setting should not be effective for a Central Instance package with the same SID.

During startup of a package `<pkg_name>`, SGeSAP checks the existence of SAP specific configuration files in the following order:

- If a `sap.conf` file exist, it is effective for compatibility reasons.
- If a `sap<pkg_name>.conf` file exist, it will overrule previous files and is effective.
- If a `sap.config` file exist, it will overrule previous files and is effective.
- If a `sap<pkg_name>.config`file exist, it will overrule previous files and is effective.

NOTE: When configuring package specific configuration files `sap<pkg_name>.config` you will usually isolate section 1 in a global sap.config file. All other sections should then be applied in the package specific configuration files `sap<pkg_name>.config`, excluding the first section.

For example SAP System C11 containing a Central Instance package with Application Server handling would then have the following SGeSAP configuration files:

- `sap.config` - Section 1,3,4 configured
- `sapciC11.config` - Section 2 configured

Optional Step: OS670

The following arrays are used to configure special treatment of ABAP Application Servers during package start, halt or failover.

These Application Servers are not necessarily virtualized or secured by Serviceguard, but an attempt can be triggered to start, stop or restart them with the package. If any triggered attempt fails, it doesn't automatically cause failure of the ongoing package operation. The attempts are considered to be non-critical. In certain setups, it is necessary to free up resources on the failover node to allow the failover to succeed. Often, this includes stopping less important SAP Systems, namely consolidation or development environments. If any of these instances is a Central Instance, it might be that there are additional Application Servers belonging to it. Not all of them are necessarily running locally on the failover node. They can optionally be stopped before the Central Instance gets shut down. This mechanism replaces the deprecated `SERVER_CONSOLIDATION` variable and the `RM*`arrays of earlier SGeSAP 3.x versions.

- In the `ASSID[*]-array` specify the SAP System IDs of the instances that should be treated with the package. Making this a configurable option allows to specify instances that belong to the clustered SAP components. It also allows specification of different SIDs for less critical SAP applications.
- In the `ASHOST[*]-array`, refer to the hosts on which the instances reside - this can be either inside or outside of the cluster.
- The `ASNAME[*]-array` holds the instance names. These names are built by the abbreviations of the services that are offered by the Application Server. The name of a Dialog instance usually is D, the name of a Central Instance often is `DVEBMGS`.
- The `ASNR[*]-array` contains the instance IDs. If the corresponding `ASHOST-entry` specifies a host that is part of the cluster, be sure to provide an ID that is different form the IDs used by any packaged instance. You should also make sure that there is no other SAP instance on the same host is using that ID.
- The `ASTREAT[*]-array` defines the way the instance is treated if the status of the package changes. There are five different values that may be configured in any combination: `START_WITH_PKG`, `STOP_WITH_PKG`, `RESTART_DURING_FAILOVER`, `STOP_IF_LOCAL_AFTER_FAILOVER`, `STOP_DEPENDENT_INSTANCES`.
  - `ASTREAT[*]=0`means that the Application Server is not affected by any changes that happens to the package status. This value makes sense to (temporarily) unconfigure the instance.
  - `${START_WITH_PKG}`: Add 1 to `ASTREAT[*]` if the Application Server should automatically be started during startup of the package.
  - `${STOP_WITH_PKG}`: Add 2 to `ASTREAT[*]` if the Application Server should automatically be stopped during halt of the package.
  - `${RESTART_DURING_FAILOVER}`: Add 4 to `ASTREAT[*]` if the Application Server should automatically be restarted if a failover of the package occurs. If the restart option is not used the SAP ABAP Engine has to be configure to use `DB-RECONNECT`.
  - `${STOP_IF_LOCAL_AFTER_FAILOVER}`: Add 8 to `ASTREAT[*]` if the Application Server should automatically be shut down if the package fails over to its node. This treatment policy will overrule the `${RESTART_DURING_FAILOVER}` value if applicable.
  - `${STOP_DEPENDENT_INSTANCES}`: Add 16 to `ASTREAT[*]` if the instance is a Central Instance and all specified Dialog Instances with the same `ASSID[]` should be stopped prior to it. This treatment policy overrules the treatment policies specified for Dialog Instances.
- For the `ASPLATFORM[*]-array` the platform for an Application Server that is controlled by the package is specified. Supported values are:
  - "HP-UX": standard Application server running on HP-UX
  - "LINUX": standard Application server running on LINUX
  - "WINDOWS": Application Server handling is not standardized as there is no standard way to open a remote DOS/Windows shell that starts SAP Application Servers on the Windows platform. sap.functions provides demo functions using the `ATAMAN` (TM) TCP Remote Logon syntax. They should be replaced by implementations in customer.functions.
  - "SG-PACKAGE": The Application Server runs as Serviceguard package within the same cluster. In this case `ASHOST` might have different values on the different package nodes. If this value is

used, you also have to specify the Dialog Instance package name in ASPKGNAME[*].The common naming conventions for Dialog Instance packages are:

- d<ASNR><SID> (new)
- app<SID><ASNR> (deprecated)

${START_WITH_PACKAGE}, ${STOP_WITH_PACKAGE} and {RESTART_DURING_FAILOVER} only make sense if ASSID[]=${SAPSYSTEMNAME}, i.e. these instances need to belong to the clustered SAP component. ${STOP_IF_LOCAL_AFTER_FAILOVER} and {STOP_DEPENDENT_INSTANCES} can also be configured for different SAP components.

The following table gives an overview of ASTREAT[*] values and their combination. The ${START_WITH_PKG}, ${STOP_WITH_PKG}, ${RESTART_DURING_FAILOVER}, ${STOP_IF_LOCAL_AFTER_FAILOVER}, ${STOP_DEPENDENT_INSTANCES} columns represent binary values and decimal values in parenthesis that are accumulated in the ASTREAT column.

Table 3-6 Overview of reasonable ASTREAT values

ASTREAT value	STOP_DEP	STOP_LOCAL	RESTART	STOP	START	Restrictions
0	0	0	0	0	0	Should only be configured for AS that belong to the same SID
1	0	0	0	0	1 (1)	
2	0	0	0	1 (2)	0	
3	0	0	0	1 (2)	1	
4	0	0	1 (4)	0	0	
5	0	0	1 (4)	0	1 (1)	
6	0	0	1 (4)	1 (2)	0	
7	0	0	1 (4)	1 (2)	1 (1)	
8	0	1 (8)	0	0	0	Can be configured for AS belonging to same SID or AS part of other SID
9	0	1 (8)	0	0	1 (1)	Should only be configured for AS that belong to the same SID
10	0	1 (8)	0	1 (1)	0	
11	0	1 (8)	0	1 (2)	1 (1)	
12	0	1 (8)	1 (4)	0	0	
13	0	1 (8)	1 (4)	0	1 (1)	
14	0	1 (8)	1 (4)	1 (2)	0	
15	0	1 (8)	1 (4)	1 (2)	1 (1)	
24	1 (16)	1 (8)	0	0	0	Should only be configured for Instances that have different SID and have dependencies to AS

```
START=${START_WITH_PKG}
STOP=${STOP_WITH_PKG}
RESTART=${RESTART_DURING_FAILOVER}
STOP_LOCAL=${STOP_IF_LOCAL_AFTER_FAILOVER}
STOP_DEP=${STOP_DEPENDENT_INSTANCES}
```

Example 1:

The primary node is also running a non-critical Dialog Instance with instance ID 01. It should be stopped and started with the package. In case of a failover a restart attempt should be made. There is a second instance outside of the cluster that should be treated the same.

```
ASSID[0]=SG1; ASHOST[0]=node1; ASNAME[0]=D; ASNR[0]=01; ASTREAT[0]=7;
ASPLATFORM[0]="HP-UX"
```

```
ASID[1]=SG1; ASHOST[1]=extern1; ASNAME[1]=D; ASNR[1]=02; ASTREAT[1]=7;
ASPLATFORM[1]="HP-UX"
```

Example 2:

The failover node is running a Central Consolidation System QAS. It shall be stopped in case of a failover to this node.

```
ASSID[0]=QAS; ASHOST[0]=node2; ASNAME[0]=DVEBMGS; ASNR[0]=10; ASTREAT[0]=8;
ASPLATFORM[0]="HP-UX"
```

The failover node is running the Central Instance of Consolidation System QAS. There are two additional Application Servers configured for QAS, one inside of the cluster and one outside of the cluster. The complete QAS shall be stopped in case of a failover. In this case, `${STOP_DEPENDENT_INSTANCES}` is specified for the Central Instance of QAS.

```
ASSID[0]=QAS; ASHOST[0]=node2; ASNAME[0]=DVEBMGS; ASNR[0]=10; ASTREAT[0]=24;
ASPLATFORM[0]="HP-UX"

ASSID[1]=QAS; ASHOST[1]=node3; ASNAME[1]=D; ASNR[1]=11; ASTREAT[1]=8;
ASPLATFORM[1]="HP-UX"

ASSID[2]=QAS; ASHOST[2]=extern1; ASNAME[2]=D; ASNR[2]=12; ASTREAT[2]=8;
ASPLATFORM[2]="HP-UX"
```

The Central Instance is treated the same way as any of the additional packaged or unpackaged instances. Use the `ASTREAT[*]-array` to configure the treatment of a Central Instance. For Example, if a Central Instance should be restarted as soon as a DB-package fails over specify the following in `sapdb<SID>.config`:

```
ASSID[0]=SG1; ASHOST[0]=node2; ASNAME[0]=DVEBMGS; ASNR[0]=00;
ASTREAT[0]=${RESTART_DURING_FAILOVER}; ASPLATFORM[0]="SG-PACKAGE";
ASPKGNAME="ciSG1"
```

If you don't restart the SAP Instances during database failover, make sure to turn on the `DB-RECONNECT` feature within their profile. Step IS1110 will describe how to achieve that.

Installation Step: LS680

The `REM_COMM` value defines the method to be used to remotely execute commands for Application Server handling.

It can be set to ssh to provide secure encrypted communications between untrusted hosts over an insecure network.

Information on how to set up ssh for each node refer to Section Cluster Node Configuration.

Default value is `remsh`.

Example:

```
REM_COMM=remsh
```

Installation Step: LS690

Use the `AS_PSTART` variable to specify the startup procedure of the configured Application Servers.

The default value here is `AS_PSTART=1` for parallel startup and `AS_PSTART=0` for sequential startup. Be aware that setting `AS_PSTART=0` will slow down your total failover time.

Example:

```
AS_PSTART=1
```

Optional Step: OS700

If your setup consists of application servers that are significantly slower than the Central Instance host, it is possible that the Central Instance shuts down before application server shutdown is completed.

This can lead to unsafe shutdowns and Instance crash.

To be safe, specify one of the following:

```
WAIT_OWN_AS=1
```

the shutdown of all application servers takes place in parallel, but the scripts do not continue before all of these shutdown processes have come to an end.

WAIT_OWN_AS=2

if the package should also wait for all application servers to come up successfully. You have to use this value if you want to prevent the integration from temporarily opening a new process group for each application server during startup.

WAIT_OWN_AS=0

can significantly speed up the package start and stop, especially if Windows NT application servers are used. Use this value only if you have carefully tested and verified that timing issues will not occur.

Optional Step: OS710

Specify SAPOSCOL_STOP=1 if saposcol should be stopped together with each instance that is stopped.

The collector will only be stopped if there is no instance of an SAP system running on the host. Specify SAPOSCOL_START=1 to start the saposcol even with packages that don't use the implicit startup mechanism that comes with SAP instance startup, e.g. database-only packages or packages that only have a SCS, ASCS or ERS instance.

Optional Step: OS720

If several packages start on a single node after a failover, it is likely that some packages start up faster than others on which they might depend.

To allow synchronization in this case, there is a loop implemented that polls the missing resources regularly.

After the first poll, the script will wait 5 seconds before initiating the next polling attempt. The polling interval is doubled after each attempt with an upper limit of 30 seconds. Polling continues until the resource becomes available or up to a maximum of DELAY_INTERVALS polling attempts.

Optional Step OS730

If several packages start on a single node after a failover, it is likely that some packages start up faster than others on which they might depend.

To allow synchronization in this case, there is a loop implemented that polls the missing resources regularly.

Control the handling of resources.

Prior to any instance startup the SGeSAP tries to free up unused or unimportant resources to make the startup more likely to succeed. A database package only frees up database related resources, a Central Instance package only removes IPCs belonging to SAP administrators.

The following list summarizes how the behavior of SGeSAP is affected by changing the CLEANUP_POLICY parameter:

▲ lazy - no action, no cleanup of resources

- normal - removes unused resources belonging to the own system
- strict - uses HP-UX commands to free up all semaphores and shared memory segments that belong to any SAP Instance of any SAP system on the host if the Instance is to be started soon. It uses HP-UX to free up all semaphores and shared memory segments that belong to any database if the SAP database is to be started soon.

NOTE: Do not use the strict policy unless it is critical that you do. Be aware that the strict option can crash running instances of different SAP systems on the backup host. Use this value only if you have a productive system that is much more important than any other SAP system you have. In this case a switchover of the productive system is more robust, but additional SAP systems will crash.

You can also use strict policy, if your SAP system is the only one running at the site and you are low on memory. strict policy frees up more of its own shared memory segments than the normal policy does.

For the cleanup of resources cleanipc is used which is an SAP tool to free up the IPC resources of specific SAP Instances. It is used to free up resources of an Instance that is to be started soon on HP-UX. This prevents shared memory issues caused by the remains of a prior crash of the Instance. Accordingly, an obsolete ORACLE SGA is also removed if a database crash occurred.

Optional Parameters and Customizable Functions

In /etc/cmcluster/<SID> there is a file called customer.functions that provides a couple of predefined function hooks. They allow the specification of individual startup or runtime steps at certain phases

of the package script execution. Therefore it is not necessary and also not allowed to change the `sap.functions`.

In the following there is a summary of the function hooks in chronological order:

Table 3-7 Optional Parameters and Customizable Functions List

Command:	Description:
`start_addons_predb`	additional startup steps on database host after all low level resources are made available after the system has been cleaned up for db start before start of database instance
`start_addons_postdb`	additional startup steps on database host after start of the database instance
`start_addons_preci`	additional startup steps on Central Instance host after start of the database instance
`start_addons_postci`	additional startup steps on Central Instance host after start of the Central Instance before start of any Application Server Instance
`start_addons_postciapp`	additional startup steps that are performed on the Central Instance host after startup of all Application Server Instances
Equally, there are hooks for the stop procedure of packages:	
`stop_addons_preciapp`	usually this function contains actions that relate to what has been added to start_addons_postciapp
`stop_addons_preci`	relates to `start_addons_postci`
`stop_addons_postci`	relates to `start_addons_preci`
`stop_addons_predb`	relates to `start_addons_postdb`
`stop_addons_postdb`	relates to `start_addons_predb`

The customer.functions template that is delivered with SGeSAP provides several examples within the hook functions. They can be activated via additional parameters as described in the following.

Stubs for JAVA based components work in a similar fashions.

Optional Step: OS740

`SAPROUTER` is an example for an additional program that always starts on the Central Instance host.

The array `SAPROUTER_START[*]` should be set to 1 for each `saprouter` that should be started with the `CI` package. Make sure that multiple `saprouters` do not share the same `SAPROUTER_PORT`. The `SAPROUTER_HOST[*]-array` specifies where the `saprouter` should be started. These parameters are mandatory. An optional `SAPROUTER_STRING` may be set that will be passed to the `saprouter`, for example to set the path to the `saprouttab` file to reside on a shared volume.

Example:

```
SAPROUTER_START[0]=1
SAPROUTER_PORT[0]="-S3299"
SAPROUTER_HOST[0]=hpcc006
SAPROUTER_STRING[0]="-W 20000 -R /sapmnt/SG1/profile/saprouttab3299 -T
sapmnt/SG1/profile/dev_rout3299"
```

The `saprouter`(s) get(s) started in the script `customer.functions (start_addons_postciapp)` and get(s) halted sequentially during package shutdown (`stop_addons_preciapp`).

Optional Step: OS750

Specify `RFCADAPTER_START=1` to automatically start the RFC adapter component,

For Example, some early Exchange Infrastructure versions. Make sure that the JVM executables can be reached via the path of `SIDADM`.

Example:

```
RFCADAPTER_START=1
RFCADAPTER_CMD="run_adapter.sh"
```

Optional Step: OS760

Specify *SAPCCMSR_START=1* ***if there should be a CCMS agent started on the DB host automatically. You should also specify a path to the profile that resides on a shared*** *lvol*

For example:

```
SAPCCMSR_START=1
SAPCCMSR_CMD=sapccmsr


SAPCCMSR_PFL="/usr/sap/ccms/${SAPSYSTEMNAME}_${CINR}/sapccmsr/${SAPSYSTEMNAME}.pfl
```

Global Defaults

The fourth section of sap.config is rarely needed. It mainly provides various variables that allow overriding commonly used default parameters.

Optional Step: OS770

If there is a special demand to use values different from the default, it is possible to redefine some global parameters.

Depending on your need for special HA setups and configurations have a look at those parameters and their description in the SAP specific configuration file `sap.config`.

When using the HP Serviceguard Storage Management Suite with CFS and shared access Option 3 (Chapter 2 - Planning the Storage Layout) it is required to set the `CTEXERELOC` variable , because otherwise the default initialization value would be `NFSRELOC` (`DEFAULT: ${NFSRELOC}`). `CTEXERELOC` should get initialized to `DBRELOC` or `CIRELOC`.

Optional Step for Netweaver 2004s: NW04S780

This installation step only applies if Netweaver 2004s with SAP Application Server 7.0 is installed.

SAP introduced a control framework with Netweaver 2004s and SAP Application Server 7.0 that enables the management of SAP instances from third party products using a defined application interface. This control framework is implemented using a daemon process that is either started by:

- the operating system during boot time or
- the SAP startup script when the instance is started.

During the installation of a SAP Netweaver 2004s Web AS, the SAPlinst installer edits a file in `/usr/sap/sapservices` and adds a line for each `sapstartservice` instance. During boot time an init script is executed located in:

```
/sbin/init.d/sapinit referenced by /sbin/rc3.d/S<###>sapinit
```

The sapinit script reads the content in file `/usr/sap/sapservices` and starts a `sapstartsrv` for each instance during OS boot time. This can lead to error output in the boot messages log file as file systems for exclusive shared storage is not available at that time.

If the use of the SAP control framework is not required, then remove the reference link from the sapinit script. Furthermore, any running `sapstartsrv` processes can be killed from the process list. For example:

```
# rm /sbin/rc3.d/S<###>sapinit
# ps -ef | grep sapstartsrv# kill {PID of sapstartsrv }
```

To make use of the `sapstartservice` configure the `SAPSTARTSRV_START` and `SAPSTARTSRV_STOP` values. The `sapstartsrv` daemon is started and stopped accordingly. For example:

```
SAPSTARTSRV_START=1
SAPSTARTSRV_STOP=1
```

Optional Step: OS790

To enable the Workload Management (WLM) support of the dispatcher monitors specify: *WLM_PROCMAP=1*

The following example WLM configuration file illustrates how to favour dialog to batch processing.

```
prm {
 groups = OTHERS : 1,
 Batch : 2,
 Dialog : 3,
 SAP_other: 4;
users = <sid>adm: SAP_other,  ora<sid>: SAP_other;
 #
 # utilize the data provided by sapdisp.mon to identify batch and dialog
 #
procmap = Batch : /opt/wlm/toolkits/sap/bin/wlmsapmap -f
/etc/cmcluster/<SID>/wlmprocmap.<SID>_<INSTNAME><INR>_<HOSTNAME> -t BTC,

Dialog :
/opt/wlm/toolkits/sap/bin/wlmsapmap -f /etc/cmcluster/P03/wlmprocmap.P03_DVEBMGS00_cutst144 -t
DIA;}

#
# definition of Service Level Objectives
#
slo s_Dialog {
pri = 1;
 entity = PRM group Dialog;
 mincpu = 1;
 maxcpu = 600;
 goal = usage _CPU;
 condition = metric sap_db_active;
}

slo s_Batch {
 pri = 2;
 entity = PRM group Batch;
 mincpu = 1;
 maxcpu = 600;
 goal = usage _CPU;
 condition = metric sap_db_active;
}
```

The following example WLM configuration file ties the CPU core shares guaranteed for dialog processing to the number of dialog processes running as of the current SAP operation mode. In times of high load, the overall dialog processing power of the instance is guaranteed to be at least 25% of a core multiplied by the number of dialog work processes running.

```
#
# Uses absolute CPU units so that 100 shares == 1 CPU.
#
tune {
 wlm_interval = 5;
 absolute_cpu_units = 1;
}

#
# Create a workload group for the dialog processes and use the wlmsapmap
# utility to move all the dialog processes to this workload group.
#
prm {
 groups = OTHERS : 1, dialog : 2;
 procmap = dialog : /opt/wlm/toolkits/sap/bin/wlmsapmap
 -f /etc/cmcluster/C11/wlmprocmap.C11_D01_node1 -t DIA; }
```

```
#
# Request 25 shares per dialog job.
#
slo s_dialog {
 pri = 1;
 entity = PRM group dialog
 cpushares = 25 shares total per metric m_dialog; }

#
# Report the number of active processes in the dialog workload group.
#
tune m_dialog {
 coll_argv = glance_prm APP_ACTIVE_PROC dialog; }
```

Optional Step: OS795

Non-default SAP parameters that can be defined include *EXEDIR* ***for the SAP access path to the SAP kernel executables (default:*** */usr/sap/<SID>/SYS/exe/run***) and*** *JMSSERV_BASE* ***(default:3900) for the offset from which the message server port of JAVA SCS instances will be calculated as*** *JMSSERV_BASE+<INSTNR>****.***

For example:

EXEDIR for a Unicode system:

```
EXEDIR= /usr/sap/<SID>/SYS/exe/runU
```

JMSSERV_BASE for most XI add-in installations:

```
JMSSERV_BASE=3600
```

HA NFS Toolkit Configuration

The cross-mounted file systems need to be added to a package that provides HA NFS services. This is usually the (j)db(j)ci package, the (j)db package or the standalone sapnfs package. Logon as root to the primary host:

NOTE: This installation section does not need to be performed when using the HP Serviceguard Storage Management Suite with CFS and shared access Option 3 (Chapter 2 - Planning the Storage Layout)

Installation Step: LS530

If it is intended to use a standalone HA NFS package, create a directory for the *sapnfs* ***Serviceguard package for the configuration and control files similar to the Installation Steps as mentioned in chapter Serviceguard Configuration.***

Refer to the documentation Managing High Available NFS for detailed instructions and configuration steps.

The naming convention for a sapnfs package is:

```
PACKAGE_NAME sapnfs
```

Installation Step: LS530

If the HA NFS functionality is intended to be integrated into a SGeSAP package, copy the HA NFS toolkit scripts into the package directory.

Since the SGeSAP package directory can have entries for both, a database and a Central Instance package, it is required to add a package type suffix to the NFS toolkit files during the copy operation. Otherwise, all packages of the package directory would act as NFS server. This is usually not intended. In standard setups, only the combined DBCI package or the dedicated SAPNFS package or the DB package needs NFS server functionality.

In a package directory, only one package of any of these package types should exist. Renaming is mandatory when using multiple package types. The monitoring script nfs.mon does not need to be renamed.

The following is an example of copy operations, whereas <pkg_type> can be db, dbci or sapnfs:

```
cp /opt/cmcluster/nfs/hanfs.sh /etc/cmcluster/<SID>/hanfs.<pkg_type>
```

In the package control script <pkg_type>.control.script, specify :

HA_NFS_SCRIPT_EXTENSION=<pkg_type>

This will allow only the package control script of the <pkg_type>.control.script to execute the HA NFS script.

Installation Step: LS540

The following steps will customize the hanfs.<pkg_type> scripts. It will customize all required directories for the usage of the HA-NFS.

All directories that are handled by the automounter must be exported by the scripts if they are part of the packages. Exported directories can usually be found beneath the special export directory /export for storage option 1 of Chapter 2. There are no cross-mounts via /export required for option 2.

Example for option 1:

/export/sapmnt/<SID>

/export/usr/sap/trans

Example for option 2:

/sapmnt/<SID>

/usr/sap/trans

Exported directories can usually be found beneath the special export directory /export. The directories to be exported are specified including their export options, using the XFS[*] array the hanfs.<pkg_type> script out of the HA-NFS toolkit. This script is called within the runtime by the standard Serviceguard control script

The transport directory is also part of the package in standard installations. In those cases it also has to be mentioned here.

Only allow access from machines inside of the cluster that are configured as backup hosts and additional application servers of this specific SAP system. Make sure to allow access for all addresses that the servers use so they can reach the directories. Allowing the right addresses is essential if your hosts use more than one LAN card.

Example of an hanfs.db file using cross-mounts with storage layout option 1:

```
XFS[0]="-o root=node1:node2:node3:trans:db:sap /export/usr/sap/trans"
XFS[1]="-o root=node1:node2:node3:trans:db:sap /export/sapmnt/<SID>"
```

Installation Step: LS550

A Serviceguard service monitor for the HA NFS can be configured in section NFS MONITOR in hanfs.<pkg_type>. The naming convention for the service should be <pkgtype><SID>NFS. Example for SAP System C11:

```
NFS_SERVICE_NAME[0]="dbciC11nfs"
NFS_SERVICE_CMD[0]="/etc/cmcluster/C11/nfs.mon"
NFS_SERVICE_RESTART[0]="-r 0"
```

Copy the monitor file nfs.mon from the HA NFS staging area into the package directory.

Auto FS Configuration

This section describes the configuration of the HP-UX automount feature that will automatically mount NFS file systems.

NOTE: This section only needs to be performed when using storage layout option 1.

Repeat the steps in this section for each node of the cluster and for each external application server host.

Logon as root.

Installation Step: IS800

Check that the automounter is active.

In /etc/rc.config.d/nfsconf, the section for the autofs configuration should look similar to this:

```
AUTOMOUNT=1
```

```
AUTO_MASTER="/etc/auto_master"
AUTOMOUNT_OPTIONS="-f $AUTO_MASTER"
AUTOMOUNTD_OPTIONS= -L
AUTOFS=1
```

Older installations on HP-UX and installations without autofs require a slightly different syntax for the "old" automounter:

```
AUTOMOUNT=1
AUTO_MASTER="/etc/auto_master"
AUTO_OPTIONS="-f $AUTO_MASTER"
```

Installation Step: IS810

Make sure that at least one NFS client daemon and one NFS server daemon is configured to run.

This is required for the automounter to work. Check the listed variables in `/etc/rc.config.d/nfsconf`. They should be specified as greater or equal to one.

For example:

```
NFS_CLIENT=1
NFS_SERVER=1
NUM_NFSD=4
NUM_NFSIOD=4
```

Installation Step: IS820

Add the following line to your *`/etc/auto_master file`****:***

```
/- /etc/auto.direct
```

Installation Step: IS830

Create a file called *`/etc/auto.direct`****.***

Identify the HA file systems as named in chapter Planning the Storage Layout and tables referring to the used database for each directory configured to be mounted below `/export`. For each HA file system add a line to `/etc/auto.direct`.

For example:

```
/usr/sap/trans <relocdbci_s>:/export/usr/sap/trans
/sapmnt/<SID> <relocdbci_s>:/export/sapmnt/<SID>
```

Add database specific filesystems to this files if they exist. This can be verified using the table in Chapter 2.

Usually the relocatible IP address of the database package or the `SAPNFS` package is used.

When configuring `AUTOFS` the automounter map file `/etc/auto.direct` must NOT be executable. Make sure to set the appropriate permissions of `/etc/auto.direct to 644`.

Installation Step: IS840

Restart the automounter with: *`/sbin/init.d/nfs.client stop`* ***and*** *`/sbin/init.d/nfs.client start`*

WARNING! Never kill the automount process. Always use `nfs.client` to stop or start it.

WARNING! Never stop the NFS client while the automounter directories are still in use by some processes. If nfs.client stop reports that some filesystems could not be unmounted, the automounter may refuse to handle them after `nfs.client` start.

Database Configuration

This section deals with additional database specific installation steps and contains the following:

- Additional Steps for Oracle
- Additional Steps for MAXDB

Additional Steps for Oracle

The Oracle RDBMS includes a two-phase instance and crash recovery mechanism that enables a faster and predictable recovery time after a crash.

The instance and crash recovery is initiated automatically and consists of two phases:

Roll-forward phase: Oracle applies all committed and uncommitted changes in the redo log files to the affected data blocks. Following parameters can be used to tune the roll forward phase:

- The parameter `RECOVERY_PARALLELISM` controls the number of concurrent recovery processes.
- The parameter `FAST_START_IO_TARGET` controls the time a `crash /` instance recovery may take. Use this parameter to make `crash /` instance recovery predictable.

Roll-back phase: Oracle applies information in the rollback segments to undo changes made by uncommitted transactions to the data blocks. Following parameters can be used to tune the roll-back phase:

- Fast-Start On-Demand rollback: with this feature Oracle automatically allows new transactions to begin immediately after the roll forward phase of recovery completes. This means that the database will be available again right after the completion of phase one roll-forward. This means that there will be no long waits until long running transactions are rolled back.
- Fast-Start Parallel Rollback: configure the F`AST_START_PARALLEL_ROLLBACK` parameter to roll-back set of transaction in parallel. This parameter is similar to the `RECOVERY_PARALLELISM` parameter for the roll-forward phase.

All these parameters can be used to tune the duration of Instance / Crash recovery.

The following steps have to be performed in order to adjust the Oracle RDBMS setting to the HA configuration.

Logon as root to the primary host of the database where the package is running in debug mode.

Oracle Database Step: OR850

Perform the following step as *<sid>adm*

To ensure that a database that crashed during an online backup starts correctly after the crash, all data files that were in `begin backup` state need to be altered with an `end backup` statement. Adjust the required steps in `/sapmnt/<SID>/exe/startdb`.

Change the following code within the `/sapmnt/<SID>/exe/startdb` file. The sample code that is to be inserted for different Oracle releases can be found in the files:
`/opt/cmcluster/sap/SID/startdb.sqlplus` and `/opt/cmcluster/sap/SID/startdb.svrmgrl`

```
#
# Startup the database without changing the ARCHIVELOG state
#
echo "connect internal;" > $SRVMGRDBA_CMD_FILE
echo "startup;" >> $SRVMGRDBA_CMD_FILE
echo "exit;" >> $SRVMGRDBA_CMD_FILE
eval $SRVMGRDBA command=@$SRVMGRDBA_CMD_FILE >> $LOG 2>&1
```

Oracle Database Step: OR860

Perform the following steps as *ora<sid>.*

Configure the listener to listen on the relocatible name of the database package. To do this, change all references from `<local>` to the relocatible name `<relocdb>` in the files on the shared volume group. Be careful if these files were customized after the SAP installation. For example:

`$ORACLE_HOME/network/admin/listener.ora`

`$ORACLE_HOME/network/admin/tnsnames.ora`

When using Oracle 10g:

`/sapmnt/<SID>/profile/oracle/listener.ora`

`/sapmnt/<SID>/profile/oracle/tnsnames.ora`

Oracle Database Step: OR870

Copy *$ORACLE_HOME/network/admin/tnsnames.ora* ***to all additional application server hosts.***

Be careful if these files were customized after the SAP installation.

Oracle Database Step: OR880

Be sure to configure and install the required Oracle NLS files and client libraries as mentioned in section Oracle Storage Considerations included in chapter Planning the Storage Layout.

Also refer to SAP OSS Note 180430 for more details.

Optional Step: OR890

If you use more than one SAP system inside of your cluster...

It is possible that more than one database is running on the same node. Even though one listener is capable of serving many database instances problems can occur in switchover environments because needed file systems may not be available at the startup time of the listener. Use a dedicated listener process for each database.

You can use the standard file that is created during the installation of one SAP system `<SID1>` as a template. Double its contents, for example:

```
cat listener.ora listener.ora >listener.ora.new
mv listener.ora.new listener.ora
```

The file consists of two identical parts

Table 3-8 Working with the two parts of the file

Part of the file:	Instruction:
first part	Replace each occurrence of the word `LISTENER` by a new listener name. You can choose what suits your needs, but it is recommended to use the syntax `LISTENER<SID1>`: `( host = <relocdb_1> )` Change nothing.

second part	Replace each occurrence of the word `LISTENER` by a new listener name different from the one chosen above. For example, use `LISTENER<SID2>` if `<SID2>` is the `SID` of the second SAP system. Replace any other occurrence of `<SID1>` by `<SID2>`. The line should be modified to contain the appropriate relocatible address belonging to the database package (`db`, `dbci` or `jdb`) of the second system. For example:
	`( host = <relocdb_2> )`
	In the line:
	`( port = 1527 )`
	a new previously unused port should be placed. For example:
	`( port = 1528 )`
	Adapt the (`host=...`) and (`port=...`) lines corresponding to the values you have chosen in the `listener.ora` file.

Test your setup by starting the listeners as `ora<sid1/2>`:

```
lsnrctl start LISTENER<SID1/2>
```

Create an `/etc/services` entry for the new port you specified above. Use `tlisrv<SID2>` as service name. The name is not needed anyhow. This entry has to be made on all hosts that run an instance that belongs to the system. This includes all external application server hosts outside of the cluster.

Optional Step: OR900

If you use multiple packages for the database and SAP components...

Set the optional parameter `SQLNET.EXPIRE_TIME` in sqlnet.ora to a reasonable value in order to take advantage of the Dead Connection Detection feature of Oracle. The parameter file `sqlnet.ora` resides either in `/usr/sap/trans` or in `$ORACLE_HOME/network/admin`.

The value of `SQLNET.EXPIRE_TIME` determines how often (in seconds) `SQL*Net` sends a probe to verify that a client-server connection is still active. If the Central Instance switches, the application servers may crash, thereby leaving shadow processes running on the database host. While the Central Instance package cleans up the application server hosts, it does not touch the ORACLE shadow processes running on the database host. Remove them, because their number increases with every Central Instance package switch. After an application server crash, a connection to the database shadow process may be left open indefinitely. If the `SQLNET.EXPIRE_TIME` parameter is specified,`SQL*Net` sends a probe periodically to determine

whether there is an invalid connection that should be terminated. It finds the dead connections and returns an error, causing the server process to exit.

Oracle Step: OR940

Additional steps for Oracle 9i RDBMS:

Some Oracle 9i the Oracle Installers create symbolic links in the client library directory `/oracle/client/92x_64/lib` that reference to SID-specific libraries residing in the `$ORACLE_HOME/lib` of that database instance. This causes trouble when the database package is not running locally - these symbolic links can not be resolved. Therefore these links are required to be resolved, i.e. copied locally in the client library directory.

But only copying the client libraries locally does not solve the issue completely: certain client libraries are linked during installation so that the 'shared dynamic path search' contains the following entries (chatr output):

```
LD_LIBRARY_PATH enabled first
SHLIB_PATH enabled second
embedded path enabled third /oracle/<SID>/920_64/lib
```

This means if the libraries are not found in `LD_LIBRARY_PATH` or `SHLIB_PATH`, the dynamic loader again searches in the instance specific path `$ORACLE_HOME/lib`. To resolve this issue, the needed libraries have to be provided either in `LD_LIBRARY_PATH` or `SHLIB_PATH` as local copies or symbolic links. For example create symbolic links to these libraries in the central executable directory `/sapmnt/<SID>/exe` is a possible solution.

Oracle Step: OR941

Additional steps for Oracle 10g RDBMS:

Even though no cluster services are needed when installing a Oracle 10g Single Instance, the cluster services daemon (`$ORACLE_HOME/bin/ocssd.bin`) is installed. It will remain running, even when the database is shut down. This keeps the file system busy during package shut down. Therefore it is mandatory that the cluster services daemon is disabled.

Comment or remove the following entry in `/etc/inittab`:

```
# h1:3:respawn:/sbin/init.d/init.cssd run >/dev/null 2>&1 </dev/null
```

Additionally there is a init script in `/sbin/init.d`:

```
/sbin/init.d/init.cssdthat is linked to by /sbin/rc3.d/S<###>init.cssd
```

and

```
/sbin/rc3.d/K<###>init.cssd
```

Remove the following referenced links to prevent the Oracle Cluster Services Daemon from initiating during boot time.

```
# rm /sbin/rc3.d/S<###>init.cssd
# rm /sbin/rc3.d/K<###>init.cssd
```

Additional Steps for MAXDB

Logon as root to the primary host of the database where the `db` or `dbci` package is running in debug mode.

MAXDB Database Step: SD930

If environment files exist in the home directory of the MAXDB user on the primary node, create additional links for any secondary.

For example:

```
su - sqd<sid>
ln -s .dbenv_<primary>.csh .dbenv_<secondary>.csh
ln -s .dbenv_<primary>.sh .dbenv_<secondary>.sh
exit
```

MAXDB Database Step: SD950

Make sure that /usr/spool exists as a symbolic link to /var/spool on all cluster nodes on which the database can run.

MAXDB Database Step: SD950

Configure the XUSER file in the <sid>adm user home directory.

The XUSER file in the home directory of the SAP Administrator keeps the connection information and grant information for a client connecting to the SAPDB database. The XUSER content needs to be adopted to the relocatable IP the SAPDB RDBMS is running on. To list all mappings in the XUSER file, run the following command as <sid>adm :

```
# dbmcli -ux SAPR3,SAP -ul
```

for MAXDB 7.6:

```
su - <sid>adm
dbmcli -ux SAP<SID>, <password> -ul
```

This command produces a list of SAPDB user keys that may be mapped to the SAPDB database schema via a local hostname. SAP created keys commonly include c, w and DEFAULT. If a mapping was created without specifying a key, entries of the form <num><SAPDBSID><hostname>, e.g. 1SDBrhpcc072 exist. These will only work on one of the cluster hosts.

To find out if a given user key mapping <user_key> works throughout the cluster, the relocatable address should be added to the primary host using cmmodnet -a.

For all previously listed user keys run the following command as <sid>adm Administrator:

```
# dbmcli -U <user_key>
```

quit exits the upcoming prompt:

```
# dbmcli on <hostname> : <SAPDBSID>> quit
```

<hostname>should be relocatable. If it is not, the XUSER mapping has to be recreated.

For example:

```
# dbmcli -d <SAPDBSID> -n <reloclc_s> -us <user>,<passwd> \ -ux SAPR3,SAP -uk
<user_key>
```

Entries of the form <num><SAPDBSID><reloclc_s> could be created by omitting the -ukoption.

NOTE: Refer to the SAP documentation to learn more about the dbmcli syntax.

After recreation always check the connection:

```
# dbmcli -U <user_key>
# dbmcli on <reloclc_s> : <SAPDBSID>> quit
```

MAXDB Database Step: SD960

Distribute the XUSER mappings by distributing the file /home/<sid>adm/.XUSER.<version> to all database package nodes.

MAXDB Database Step: SD965

When using raw devices, the database installation on the primary node might have modified the access rights and owner of the device file entries where the data and logs reside.

These devices were changed to be owned by user sdb, group sdba. These settings also have to be applied on the secondary, otherwise the database will not change from ADMIN to ONLINE state.

SAP Application Server Configuration

This section describes some required configuration steps that are necessary for SAP to become compatible to a cluster environment. The following configuration steps do not need to be performed if the SAP System was installed using virtual IPs. The steps are only required to make a non-clustered installation usable for clustering.

- SAP ABAP Engine specific configuration steps
- SAP J2EE Engine specific configuration steps

SAP ABAP Engine specific configuration steps

Logon as <sid>adm on the primary node on which a Central Instance or the System Central Services have been installed. The appropriate Serviceguard package should still run on this host in debug mode.

Installation Step: IS1100

For SAP Central Instance virtualization, change into the profile directory by typing the alias:

```
cdpro
```

In the DEFAULT.PFL change the following entries and replace the hostname with the relocatible name if you cluster a ci component. For example:

```
SAPDBHOST = <relocdb>
rdisp/mshost = <relocci>
rdisp/sna_gateway = <relocci>
rdisp/vbname = <relocci>_<SID>_<instnr>
rdisp/btcname = <relocci>_<SID>_<instnr>
rslg/collect_daemon/host = <relocci>
```

If you don't have Replicated Enqueue:

```
rdisp/enqname = <relocci>_<SID>_<instnr>
```

The following parameters are only necessary if an application server is installed on the adoptive node. For example:

```
rslg/send_daemon/listen port
rslg/collect_daemon/listen port
rslg/collect_daemon/talk port
```

Optional Step: IS1110

If you want to use the DB-Reconnect functionality for ABAP Instances:

Add entries to:

- /sapmnt/<SID>/profile/<SID>_<INSTNAME><INR>
- Instance Profiles of all application servers that use DB-Reconnect

For example:

```
rsdb/reco_trials = 15
rsdb/reco_sleep_time = 60
rsdb/reco_sosw_for_db = off
```

(based on OSS #109036)

```
rsdb/reco_sync_all_server = on
```

Installation Step: IS1120

In the instance profile of each clustered ABAP instance, two values have to be added to specify the virtual addressing,

For example in <SID>_DVEBMGS<INSTNR>:

```
SAPLOCALHOST = <relocci>
SAPLOCALHOSTFULL = <relocci>.<domain>
```

The parameter SAPLOCALHOSTFULL must be set even if you do not use DNS. In this case you should set it to the name without the domain name:

```
SAPLOCALHOSTFULL=<relocci>
```

The instance profile name is often extended by the hostname. You do not need to change this filename to include the relocatible hostname. SGeSAP also supports the full instance virtualization of SAPWAS 6.40 and beyond. The startsap mechanism can then be called by specifying the instance's virtual IP address. For this case, it is required to use the virtual IP addressing in the filenames of the instance profile and instance start profile. All references to the instance profiles that occur in the start profile need to be changed to include the virtual IP address instance profile filename.

SAPLOCALHOST is set to the hostname per default at startup time and is used to build the SAP application server name:

<SAPLOCALHOST>_<SID>_<INSTNR>

This parameter represents the communication path inside an SAP system and between different SAP systems. SAPLOCALHOSTFULL is used for rfc-connections. Set it to the fully qualified hostname.

The application server name appears in the server list held by the message server, which contains all instances, hosts and services of the instances. The application server name or the hostname is also stored in some system tables on the database.

Installation Step: IS1130

If you cluster a single-instance database, you need to maintain the transport management parameters.

In the transport directory parameter file(s), e.g. /usr/sap/trans/bin/TPPARAM, modify the dbhost entry as follows:

<SID>/dbhost = <relocdb>

Optional Step: IS1140

If you have already received your licenses from SAP install them on all the nodes where the Central Instance can start.

Refer to the Serviceguard Extension for SAP Release Notes for further information on how to do this. The package comes up without a license, too. But certain restrictions apply to the SAP application.

SAP will now be ready to run on all cluster nodes. Test the manual startup on all adoptive nodes.

Installation Step: IS1150

Connect with a SAPGUI. Import the changed SAP profiles within SAP.

The transaction is called RZ10.

After importing the profiles, check with rsparam in SE38 if the parameters SAPLOCALHOST and SAPLOCALHOSTFULL are correct. If you do not import the profiles, the profiles within SAP can be edited by the SAP Administrator. The values listed from the SAP system will be wrong and, when saved, will overwrite the values which edited on the HP-UX level.

Installation Step: IS1160

The destination for print formatting, which is done by a Spool Work process, uses the application server name.

If the application server name stays the same because SAPLOCALHOST has been set to the relocatible name, after a switch no changes need to be done.

Printing works consistently. A print job in process at the time of the failure is canceled and needs to be reissued manually after the failover. To make a spooler highly available on the Central Instance server, set the destination of the printer to e.g. <relocci>_<SID>_<INSTNR> using transaction SPAD.

Installation Step: IS1170

Batch jobs can be scheduled to run on a particular instance.

You select a particular instance by its hostname at the time of job scheduling. The application server name and the hostname retrieved from the Message Server are stored in the Batch control tables TBTCO, TBTCS . . .

When the batch job is ready to run, the application server name is used to start it on the appropriate instance. When using <relocci> to build the application server name for the SAP instance, you do not need to change batch jobs, which are tied to the locality, after a switchover, even if the hostname which is also stored in the above tables differs.

Installation Step: IS1180

Within the SAP Computing Center Management System (CCMS) you can define operation modes for SAP instances.

An operation mode defines a resource configuration for the instances in your SAP system. It can be used to determine which instances are started and stopped, and how the individual services are allocated for each instance in the configuration.

An instance definition for a particular operation mode consist of the number and types of work processes as well as start and instance profiles (starting with version 3.0 the CCMS allows profile maintenance from within the SAP system).

When defining an instance for an operation mode, enter the hostname and the system number of the application server. By using `<relocci>` to fill in the hostname field, the instance is working under control of the CCMS after a failover without any change.

If an instance is running on the standby node in normal operation (and is stopped when switching over), the control console shows this instance to be down (for example, you will get a red node on a graphical display) after the switchover.

Installation Step: IS1190

A SAP Internet Communication Manager (ICM) may run as part of any Application Server.

It is started as a separate multi-threaded process and can be restarted independently from the Application Server. E.g. usage of BW web reporting, Business Server Pages (BSP) rely on ICM to work properly. In scenarios using external web communication, one or more SAP Web Dispatcher often accompanies the ICMs. The SAP Web Dispatcher can be laid out redundantly with hardware load balancing. SAP also allows SGeSAP packaging for a single Web Dispatcher.

To find out if ICM is configured as part of an Application Server, check for the following SAP profile parameter:

`rdisp/start_icman=TRUE`

ICM gets switched over with the Instance package. In order to make it work, ICM has to be registered with the message server using a virtual IP address.

This mechanism is different from `SAPLOCALHOST` and `SAPLOCALHOSTFULL` since ICM allows HTTP Server Aliasing via `icm/server_port_<nn>` settings. During startup, `icman` reads its configuration and propagates it to the SAP Message Server or the SAP Web Dispatcher Server. These servers then act as the physical point of access for HTTP requests. They classify requests and send HTTP redirects to the client in order to connect them to the required ICM Instance.

This only works properly if the bound ICM instances propagate virtual IP addresses. Therefor a parameter

`icm/host_name_full=<relocatible_ip>`

needs to be set within the Instance Profile of the Application Server.

Double-check the settings with report `RSBB_URL_PREFIX_GET` or review the parameters within the SMMS transaction.

Installation Step: IS1200

If you cluster a Central Instance or an ABAP System Central Service Instance, configure the frontend PCs to attach to *`<relocci>`****.***

Most of the time, this can be achieved by distributing a new `saplogon.ini` file to the windows directory.

SAP J2EE Engine specific installation steps

This section is applicable for SAP J2EE Engine 6.40 based installations that were performed prior to the introduction of SAPINST installations on virtual IPs. There is a special SAP OSS Note 757692 that explains how to treat a hostname change and thus the virtualization of a SAP J2EE Engine 6.40. Depending on the SAP application component that is clustered, there might be additional steps documented as part of a SAP whitepaper that covers clustering of the component.

Installation Step: NW04J1210

If the SAP component installed also depends and utilizes a SAP J2EE Engine, some configuration tasks have to be performed in order for the J2EE instance to interact in the virtualized environment.

The following lines will give some hints about the settings necessary for the switchover of the J2EE Engine 6.40, but always refer to SAP OSS Note 757692 because this note is always kept up-to-date.

For virtualizing the J2EE DB Connection:

- Log on to the J2EE Configuration Tool
- Choose secure store
- Change the following values: -

admin/host/<SID> <relocdb>

For Oracle databases replace the hostname in the connection string:

jdbc/pool/<SID>/Url jdbc:oracle:thin@<relocdb>:1527:<SID>

Installation Step: NW04J1220

These settings have to be adjusted for the switchover of the J2EE part of the SAP WEB AS; the following configuration has to be performed in the Offline Configuration Editor:

▲ Log on to the Offline Configuration Editor.

Table 3-9 IS1130 Installation Step

Choose...	Change the following values
cluster_data -> dispatcher -> cfg -> kernel -> Propertysheet LockingManager	enqu.host = <relocci> enq.profile.filename=/sapmnt/<SID>/profile\ /<SID>_<INR>_<relocci>
cluster_data -> dispatcher -> cfg -> kernel -> Propertysheet ClusterManager	ms.host = <relocci>
cluster_data -> server -> cfg -> kernel -> Propertysheet LockingManager	enqu.host = <relocci> enq.profile.filename=/sapmnt/<SID>/profile\ /<SID>_<INR>_<relocci>
cluster_data -> server -> cfg -> kernel -> Propertysheet ClusterManager	ms.host = <relocci>
cluster_data -> Propertysheet instance.properties.IDxxxxxx	instance.ms.host localhost to <relocci>
(in file) /usr/sap/<SID>/<INSTNAME><INR>/j2ee\ a/additionalsystemproperties	com.sapl.instanceId=<SID>_<relocci>_<INR>

Check for the latest version of SAP OSS Note 757692.

Installation is complete.

The next step is to do some comprehensive switchover testing covering all possible failure scenarios. It is important that all relevant SAP application functionalities are tested on the switchover nodes. There exist several documents provided by HP or SAP that can guide you through this process.

4 SAP Supply Chain Management

Within SAP Supply Chain Management (SCM) scenarios two main technical components have to be distinguished: the APO System and the liveCache. An APO System is based on SAP Application Server technology. Thus, `sgesap/sapinstance` and `sgesap/dbinstance` modules can be used or `ci, db, dbci, d` and `sapnfs` legacy packages may be implemented for APO. These APO packages are set up similar to Netweaver packages. There is only one difference. APO needs access to liveCache client libraries. These files usually are searched via the path `/sapdb/programs`. How access to the client libs is ensured depends on the setup of the liveCache and is discussed in combination with the liveCache volume group layout below.

The second technical SCM component, called liveCache, is an in-memory database. The technical realization of liveCache is based on MAXDB technology. Even though liveCache is somewhat similar to the SAP owned database technologies, it needs special treatment. This is because a liveCache instance usually is closely coupled with the accompanying APO. SAP has defined a list of policies and requirements to failover solutions in order to allow support for liveCache High Availability.

Table 4-1 Supported SGeSAP lc package types

Master Token Failover with hot standby system (hss)	Instance Failover and Restart
Use this solution for those more demanding environments that require failover times in the minute range, regardless of the size of the liveCache database.	Use this solution for small liveCache databases, where the liveCache restart times during failover satisfies the service level that the business need dictates.Take into account that you may temporarily see a significant performance degradation as the liveCache is restarted.

This chapter describes how to configure and setup SAP Supply Chain Management using a liveCache cluster. The `sgesap/livecache` module and the `lc` legacy package type are explained in the context of failover and restart clusters as well as hot standby architectures.

Topics are as follows:

- Planning the Volume Manager Setup
- HP-UX Setup
  - Cluster Node Synchronization
  - Cluster Node Configuration
- SGeSAP Package Configuration
  - Modular Package Creation
  - Legacy Package Creation
  - Service Monitoring
- APO Setup Changes
- General Serviceguard Setup Changes

The tasks are presented as a sequence of steps. Each mandatory installation step is accompanied by a unique number of the format XXnnn, where nnn are incrementing values and XX indicates the step relationship, as follows:

- LCnnn—LP Package Installation Steps
- GSnnn—General Installation Steps that manipulate already existing SGeSAP integrations

Whenever appropriate, HP-UX sample commands are given to guide you through the process in as detailed a manner as possible. It is assumed that hardware as well as the operating system and Serviceguard are already installed properly on all cluster hosts. Sometimes a condition is specified with the installation step. Follow the information presented only if the condition is true for your situation.

NOTE: For installation steps in this chapter that require the adjustment of SAP specific parameter in order to run the SAP application in a switchover environment usually example values are given. These values are

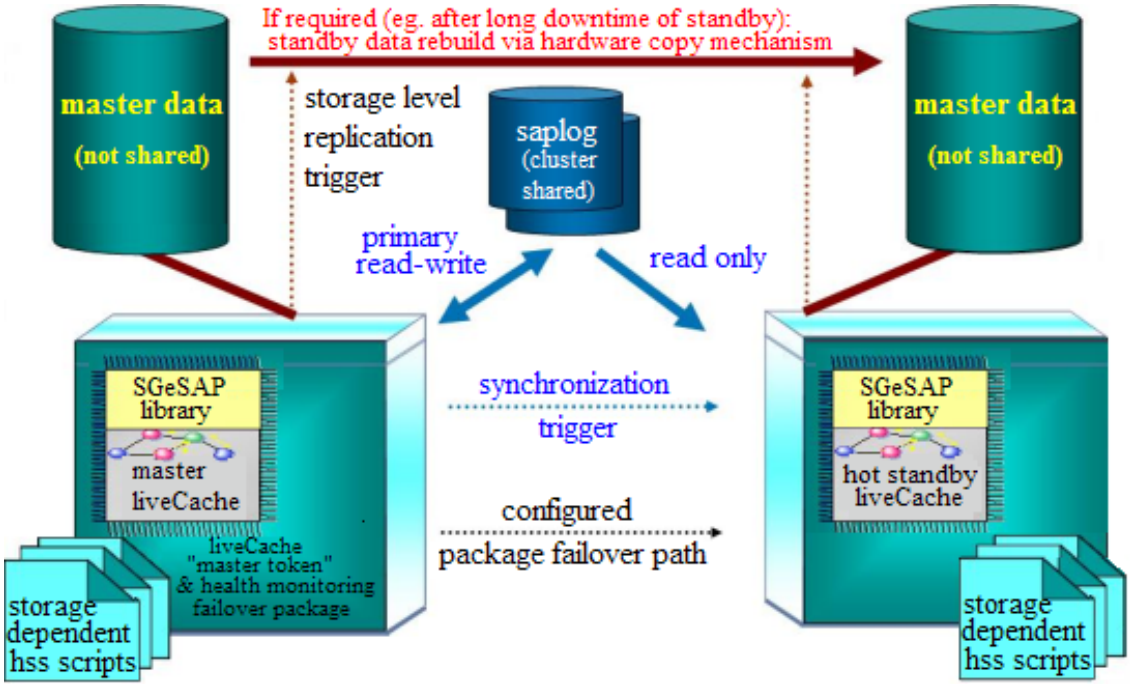
for reference ONLY and it is recommended to read and follow the appropriate SAP OSS notes for SAP's latest recommendation. Whenever possible the SAP OSS note number is given.

More About Hot Standby

A fatal liveCache failure concludes in a restart attempt of the liveCache instance. This will take place either locally or, as part of a cluster failover, on a remote node. It is a key aspect here, that liveCache is an in-memory database technology. While the bare instance restart as such is quick, the reload of the liveCache in-memory content can take a significant amount of time, depending on the size of the liveCache data spaces. Modern Supply Chain Management (SCM) business scenarios cannot afford unplanned loss of liveCache functionality for the span of time that it takes to reload the liveCache after failover. The situation becomes worse, since a restarted liveCache requires additional runtime to gain back its full performance. In many cases, systems that are connected to the liveCache can't continue with the information exchange and the outbound queues run full. The communication gets stuck until a manual restart of the queues is triggered.

The requirement to enable even the most mission- and time-critical use-cases of SCM triggered the introduction of the hot standby liveCache system (hss). Refer ti figure 4-1.

Figure 4-1 Hot Standby liveCache



A hot standby liveCache is a second liveCache instance that runs with the same System ID as the original master liveCache. It will be waiting on the secondary node of the cluster during normal operation. A failover of the liveCache cluster package does not require any time consuming filesystem move operations or instance restarts. The hot standby simply gets notified to promote itself to become the new master. The Serviceguard cluster software will make sure that the primary system is shut down already in order to prevent a split-brain situation in which two liveCache systems try to serve the same purpose. Thus, a hot standby scenario provides extremely fast and reliable failover. The delay caused by a failover becomes predictable and tunable. No liveCache data inconsistencies can occur during failover.

The hot standby mechanism also includes data replication. The standby maintains its own set of liveCache data on storage at all times.

SGeSAP provides a runtime library to liveCache that allows to automatically create a valid local set of liveCache devspace data via Storageworks XP Business Copy volume pairs (pvol/svol BCVs) as part of the standby startup. If required, the master liveCache can remain running during this operation. The copy utilizes fast storage replication mechanisms within the storage array hardware to keep the effect on the running master liveCache minimal. Once the volume pairs are synchronized, they get split up immediately. During normal operation, each of the two liveCache instances operates on a set of LUNs in SIMPLEX state.

The detection of volumes that need replication as part of the standby startup is dynamically identified within the startup procedure of the standby. It does not require manual maintenance steps to trigger volume pair synchronizations and subsequent split operations. Usually, synchronizations occur only in rare cases, for example for the first startup of a standby or if a standby got intentionally shut down for a longer period of time. In all other cases, the liveCache logging devspaces will contain enough delta information to update the standby data without the requirement to do hardware replications of full LUNs.

The ongoing operation of the standby as well as the master failover does not require the business copy mechanisms. The standby synchronizes the data regularly by accessing the master log files, which therefore reside on CVM/CFS volumes. No liveCache content data needs to be transferred via LAN at any point in time.

The liveCache logging gets continuously verified during operation. An invalid entry in the log files gets detected immediately. This avoids the hazardous situation of not becoming aware of corrupted log files until they fail to restore a production liveCache instance.

A data storage corruption that could happen during operation of the master does not get replicated to the only logically coupled standby. The standby LUNs used for devspace content do not necessarily keep the same data as the master LUNs. At least not on a physical level. But logically the standby keeps consistency and stays close to the content of the master LUNs. The standby can then be promoted to become the new master immediately. With access to the original log of the master, it is able to update itself without any data loss.

Planning the Volume Manager Setup

In the following, the lc package of SGeSAP gets described. The lc package was developed according to the SAP recommendations and fulfills all SAP requirements for liveCache failover solutions.

liveCache distinguishes an instance dependant path `/sapdb/<LCSID>` and two instance independent paths `IndepData` and `IndepPrograms`. By default all three point to a directory below `/sapdb`.

NOTE: `<LCSID>` denotes the three-letter database name of the liveCache instance in uppercase. `<lcsid>` is the same name in lowercase.

There are different configuration options for the storage layout and the filesystems caused by a trade-off between simplicity and flexibility. The options are described below ordered by increasing complexity. The cluster layout constraints that need to be fulfilled to allow the simplifications of a given option are stated within a bullet list.

The subsequent sections refer to the options by the numbers that are introduced here.

Option 1: Simple Clusters with Separated Packages

Cluster Layout Constraints:

- The liveCache package does not share a failover node with the APO Central Instance package.
- There is no MAXDB or additional liveCache running on cluster nodes.
- There is no intention to install additional APO Application Servers within the cluster.
- There is no hot standby liveCache system configured.

Table 4-2 File System Layout for liveCache Package running separate from APO (Option 1)

Storage Type	Package	Mount Point
shared	lc<LCSID>	/sapdb/data
shared	lc<LCSID>	/sapdb/<LCSID>/datan
shared	lc<LCSID>	/sapdb/<LCSID>/logn
shared	lc<LCSID>	/var/spool/sql
shared	lc<LCSID>	/sapdb/programs

In the above layout all relevant files get shared via standard procedures. The setup causes no administrative overhead for synchronizing local files. SAP default paths are used.

NOTE: The data and log disk spaces of MAXDB are called `devspaces`. For security and performance reasons SAP recommends to place the devspaces on raw devices.

Option 2: Non-MAXDB Environments

Cluster Layout Constraints:

- There is no MAXDB or additional liveCache running on cluster nodes. Especially the APO System RDBMS is either based on , ORACLE or DB2, but not on MAXDB.
- There is no hot standby liveCache system configured.

Often APO does not rely on MAXDB as underlying database technology. But independent from that, all Instances of the APO System still need access to the liveCache client libraries. The best way to deal with this is to make the client libraries available throughout the cluster via AUTOFS cross-mounts from a dedicated NFS package.

Table 4-3 File System Layout for liveCache in a non-MAXDB Environment (Option 2)

Storage Type	Package	Mount Point
shared	lc<LCSID>	/sapdb/data
shared	lc<LCSID>	/sapdb/<LCSID>
shared	lc<LCSID>	/sapdB/<LCSID>/datan
shared	lc<LCSID>	/sapdb/<LCSID>/logn
shared	lc<LCSID>	/var/spool/sql
autofs shared	sapnfs1	/sapdb/programs

1 This can be any standard, standalone NFS package. The SAP global transport directory should already be configured in a similar package. This explains why this package is often referred to as "the trans package" in related literature. A trans package can optionally be extended to also serve the global liveCache fileshares.

Option 3: Full Flexibility

If multiple MAXDB based components are either planned or already installed, the setup looks different. All directories that are shared between MAXDB instances must not be part of the liveCache package. Otherwise a halted liveCache package would prevent that other MAXDB instances can be started.

Cluster Layout Constraint:

▲ There is no hot standby system configured.

The following directories are affected:

`/sapdb/programs`: This can be seen as a central directory with `liveCache/MAXDB` binaries. The directory is shared between all `liveCache/MAXDB` Instances that reside on the same host. It is also possible to share the directory across hosts. But it is not possible to use different executable directories for two `liveCache/MAXDB` Instances on the same host. Furthermore, it is not unusual to install different MAXDB versions on the same host. This can be seen from the LC1/AP1 example of the `SAP_DBTech.ini` file printed below. It is a standard version combination for SCM3.1.

During normal operation, the liveCache most likely won't share the host with the APO database. But the failover might allow mutual backup between the liveCache host and the APO host. This implies that the files in `/sapdb/programs` have to be of the highest version that any MAXDB in the cluster has. Files in `/sapdb/programs` are downwards compatible. For the liveCache 7.4 and the APO 3.1 using MAXDB 7.3 this means that within `/sapdb/programs` there have to be the 7.4 version executables installed.

`/sapdb/data/config`: This directory is also shared between instances, though you can find lots of files that are Instance specific in here, e.g. `/sapdb/data/config/<LCSID>.*` According to SAP this path setting is static.

`/sapdb/data/wrk`: The working directory of the main liveCache/MAXDB processes is also a subdirectory of the IndepData path for non-HA setups. If a liveCache restarts after a crash, it copies important files from

this directory to a backup location. This information is then used to determine the reason of the crash. In HA scenarios, for liveCache versions lower than 7.6, this directory should move with the package. SAP allows you to redefine this path for each `liveCache/MAXDB` individually. SGeSAP expects the work directory to be part of the lc package. The mount point moves from `/sapdb/data/wrk` to`/sapdb/data/<LCSID>/wrk`. This directory should not be mixed up with the directory `/sapdb/data/<LCSID>/db/wrk` that might also exist. Core files of the kernel processes are written into the working directory. These core files have file sizes of several Gigabytes. Sufficient free space needs to be configured for the shared logical volume to allow core dumps.

NOTE: For liveCache starting with version 7.6 these limitations do not exist any more. The working directory is utilized by all instances (`IndepData/wrk`) and can be globally shared.

`/var/spool/sql`: This directory hosts local runtime data of all locally running `liveCache/MAXDB Instances`. Most of the data in this directory becomes meaningless in the context of a different host after failover. The only critical portion that still has to be accessible after failover is the initialization data in `/var/spool/sql/ini`. This directory is almost always very small (< 1MB).

Table 4-4 General File System Layout for liveCache (Option 3)

Storage Type	Package	Mount Point
shared	lc<LCSID>	/sapdb/<LCSID>
shared	lc<LCSID>	/sapdb/<LCSID>WRK*
shared	lc<LCSID>	/sapdB/<LCSID>/datan
shared	lc<LCID>	/sapdb/<LCSID>/saplogn
shared	sapnfs	/export/sapdb/programs
autofs shared	sapnfs	/export/sapdb/data
autofs shared	sapnfs	/var/spool/sql/ini
* only valid for liveCache versions lower than 7.6.		

Option 4: Hot Standby liveCache

Two liveCache instances are running in a hot standby liveCache cluster during normal operation. No instance failover takes place. This allows to keep instance-specific data local to each node. The cluster design follows the principle to share as little as possible.

The following directories are relevant:

Table 4-5 File System Layout for Hot Standby liveCache

Access Point	Package	Mount Point	Volume Manager
local	none	/sapdb/<LCSID>	any
local	none	/sapdb/programs	any
local	none	/sapdb/data	any
local	none	/var/spool/sql/ini	any
local; with XP BCVs between the nodes	none	/sapdb/<LCSID>/datan	must be LVM-based
shared disk and CFS	SG-CFS-DG-<LCSID>logn SG-CFS-MP-<LCSID>logn	/sapdb/<LCSID>/logn	must be CVM-based

MAXDB Storage Considerations

SGeSAP supports current MAXDB, liveCache and older MAXDB releases. The considerations made below will apply similarly to MAXDB,liveCache and MAXDB clusters unless otherwise indicated.

MAXDB distinguishes an instance dependant path /sapdb/<DBSID> and two instance independent paths, called IndepData and IndepPrograms. By default all three point to a directory below /sapdb.

The paths can be configured in a configuration file called /var/spool/sql/ini/SAP_DBTech.ini. Depending on the version of the MAXDB database this file contains different sections and settings.

A sample SAP_DBTech.ini for a host with a MAXDB 7.4 (LC1) and an APO 3.1 using a MAXDB 7.3 database instance (AP1):

```
[Globals]
IndepData=/sapdb/data
IndepPrograms=/sapdb/programs
[Installations]
/sapdb/LC1/db=7.4.2.3,/sapdb/LC1/db
/sapdb/AP1/db=7.3.0.15,/sapdb/AP1/db
[Databases]
.MAXDBLC=/sapdb/LC1/db
LC1=/sapdb/LC1/db
_MAXDBAP=/sapdb/AP1/db
AP1=/sapdb/AP1/db
[Runtime]
/sapdb/programs/runtime/7240=7.2.4.0,
/sapdb/programs/runtime/7250=7.2.5.0,
/sapdb/programs/runtime/7300=7.3.0.0,
/sapdb/programs/runtime/7301=7.3.1.0,
/sapdb/programs/runtime/7401=7.4.1.0,
/sapdb/programs/runtime/7402=7.4.2.0,
```

For MAXDB and liveCache Version 7.5 (or higher) the SAP_DBTech.ini file does not contain sections [Installations], [Databases] and [Runtime]. These sections are stored in separate files Installations.ini, Databases.ini and Runtimes.ini in the IndepData path /sapdb/data/config.

A sample SAP_DBTech.ini, Installations.ini, Databases.ini and Runtimes.ini for a host with a liveCache 7.5 (LC2) and an APO 4.1 using a MAXDB 7.5 (AP2):

from /var/spool/sql/ini/SAP_DBTech.ini:

```
[Globals]
IndepData=/sapdb/data
IndepPrograms=/sapdb/programs

from /sapdb/data/config/Installations.ini:
[Installations]
/sapdb/LC2/db=7.5.0.15,/sapdb/LC2/db
/sapdb/AP2/db=7.5.0.21,/sapdb/AP2/db

from /sapdb/data/config/Databases.ini:
[Databases]
.M750015=/sapdb/LC2/db
LC2=/sapdb/LC2/db
.M750021=/sapdb/AP2/db
AP2=/sapdb/AP2/db

from /sapdb/data/config/Runtimes.ini:
[Runtime]
/sapdb/programs/runtime/7500=7.5.0.0
```

liveCache Installation Step: LC010

If you decided to use option three, and the liveCache version is lower than 7.6:

1. Log on as <lcsid>adm on the machine on which liveCache was installed.

   Make sure, you have mounted a sharable logical volume on /sapdb/<LCSID>/wrk as discussed above.

2. Change the path of the runtime directory of the liveCache and move the files to the new logical volume accordingly.

```
cd /sapdb/data/wrk/<LCSID>
find . -depth -print | cpio -pd /sapdb/<LCSID>/wrk
cd ..
rm -r /sapdb/data/wrk/<LCSID>
dbmcli -d <LCSID> -u control,control
dbmcli on <LCSID>>param_directget RUNDIRECTORY
OK
RUNDIRECTORY /sapdb/data/wrk/<LCSID>
---
dbmcli on <LCSID>>param_directput RUNDIRECTORY /sapdb/<LCSID>/wrk
OK
---
dbmcli on <LCSID>>
```

liveCache Installation Step: LC020

Mark all shared non-CFS liveCache volume groups as members of the cluster.

This only works if the cluster services are already available. For example:

```
cd /
# umount all logical volumes of the volume group
vgchange -a n <vg_name>
vgchange -c y <vg_name>
vgchange -a e <vg_name>
# remount the logical volumes
```

The device minor numbers must be different from all device minor numbers gathered on the other hosts. Distribute the shared volume groups to all potential failover nodes.

HP-UX Setup for Options 1, 2 and 3

This section describes how to synchronize and configure the HP-UX installations on all cluster nodes in order to allow that the same liveCache instance is able to run on any of these nodes. This section does not apply to hot standby systems, since they never perform any instance failover.

For systems with hot standby this entire HP-UX setup section can be skipped.

Cluster Node Synchronization

1. Repeat the steps in this section for each node of the cluster that is different from the primary. .
2. Logon as root to the primary host and prepare a logon for each of its backup hosts.

liveCache Installation Step: LC030

Synchronize the *`/etc/group and /etc/passwd`* ***files.***

The liveCache installation has created a <lcsid>adm user belonging to the sapsys group. Make sure the user and group exist on all nodes and that UID and GID are consistent across the nodes.

liveCache Installation Step: LC040

Synchronize the *`sql<nn>`* ***entries from the*** *`/etc/services`* ***file between the nodes to match the entries on the primary node.***

liveCache Installation Step: LC050

Change the HP-UX kernel on the backup nodes to meet the SAP liveCache requirements as specified in the SAP liveCache installation documents.

liveCache Installation Step: LC060

Do the following to continue:

1. Copy the content of the <lcsid>adm home directory to the backup node. This is a local directory on each node.
2. Rename the environment scripts on the secondary nodes. Some of the environment scripts may not exist. For example:

```
su - <lcsid>adm
mv .dbenv_<primary>.csh .dbenv_<secondary>.csh
mv .dbenv_<primary>.sh .dbenv_<secondary>.sh
```

For liveCache 7.6:

```
su - <lcsid>adm
mv .lcenv_<primary>.csh .lcenv_<secondary>.csh
mv .lcenv_<primary>.sh .lcenv_<secondary>.sh
```

NOTE: Never use the relocatable address in these file names.

liveCache Installation Step: LC061

Copy file *`/etc/opt/sdb`* ***to the second cluster node.***

This file contains global path names for the liveCache instance

liveCache Installation Step: LC062

Verify that the symbolic links listed below in directory *`/var/spool/sql`* ***exist on both cluster nodes.***

```
dbspeed -> /sapdb/data/dbspeed
diag -> /sapdb/data/diag
fifo -> /sapdb/data/fifo
ini
ipc -> /sapdb/data/ipc
pid -> /sapdb/data/pid
pipe -> /sapdb/data/pipe
ppid -> /sapdb/data/ppid
```

liveCache Installation Step: LC070

Make sure *`/var/spool/sql`* ***exists as a directory on the backup node.***

`/usr/spool` must be a symbolic link to `/var/spool`.

liveCache Installation Step: LC080

On the backup node, create a directory as future mountpoint for all relevant directories from the table of section that refers to the layout option you chose.

Option 1:

```
mkdir /sapdb
```

Option 2:

```
mkdir -p /sapdb/data
mkdir /sapdb/<LCSID>
```

Option 3:

```
mkdir -p /sapdb/<LCSID>
```

Cluster Node Configuration

liveCache Installation Step: LC100

Repeat the steps in this section for each node of the cluster.

liveCache Installation Step: LC110

Add all relocatable IP address information to `/etc/hosts`. Remember to add the heartbeat IP addresses.

liveCache Installation Step: LC120

If you use DNS:

Configure `/etc/nsswitch.conf` to avoid problems. For example:

```
hosts: files[NOTFOUND=continue UNAVAIL=continue \
TRYAGAIN=continue]dns
```

HP-UX Setup for Option 4

This section describes how to install a hot standby instance on the secondary node.

hot standby Installation Step: LC143

The hot standby instance of storage option 4 can be installed with the installer routine that comes on the SAP liveCache media.

There is no special option for a hot standby provided by SAP yet, but installation is very straight-forward. "Install" or "patch" the master instance on the standby host. This creates local copies of all required directories, users, services and files. log files and data files should not be defined in this step.

hot standby Installation Step: LC145

In this step, the correct storage layout to support the hardware based copy mechanisms for the `devspaces` of the data files gets created.

On the Storageworks XP array `pvol/svol` pairs should be defined for all LUNs containing DISKDnnnn information of the master. These LUNs should be used for devspace data exclusively. Make sure that all pairs are split into SMPL (simple) state.

NOTE: A hot standby configuration requires the BCV copy mechanism to be preconfigured, but LUNs will not be in synchronization mode during normal operation.

hot standby Installation Step: LC147

Manually create the same LVM volume group layout for the data files on the secondary node as is in place on the primary.

Create temporary volume groups, logical volumes, filesystems and mountpoints by using the master instance as a reference. Instead of using the same LUNs as the master, use the corresponding `svols` that are in SMPL state. These `svol` LUNs will be synchronized later to become populated with a copy of the master data. The hot standby instance will take the LVM volume group layout created here as a reference on how to recreate its storage structure during startup.

hot standby Installation Step: LC149

The liveCache instances require full administration control to handle their storage configuration.

There is a special callback binary on the primary and secondary node that acts as a gateway to these functions. The ownership and permissions of this binary need to be manually changed on both nodes to have the root user confirm this step:

```
chmod 4750 /opt/cmcluster/sap/bin/hsscb
chown root:sdba /opt/cmcluster/sap/bin/hsscb
```

hot standby Installation Step: LC152

Add `/opt/cmcluster/sap/lib/hpux64` to the root and `<LCSID>adm SHLIB_PATH` environment variables on both nodes.

hot standby Installation Step: LC155

The XP Raidmanager needs to be installed and configured correctly on both nodes.

Unless defined differently in the root user environment, SGeSAP will default to the following settings:

```
HORCC_MRCF=1
HORCMINST=0
HORCM_CONF=/HORCM/etc/horcm0.conf
```

The Raidmanager binaries need to accessible via the PATH of root. Two HORCM instances can be configured on each node

hot standby Installation Step: LC157

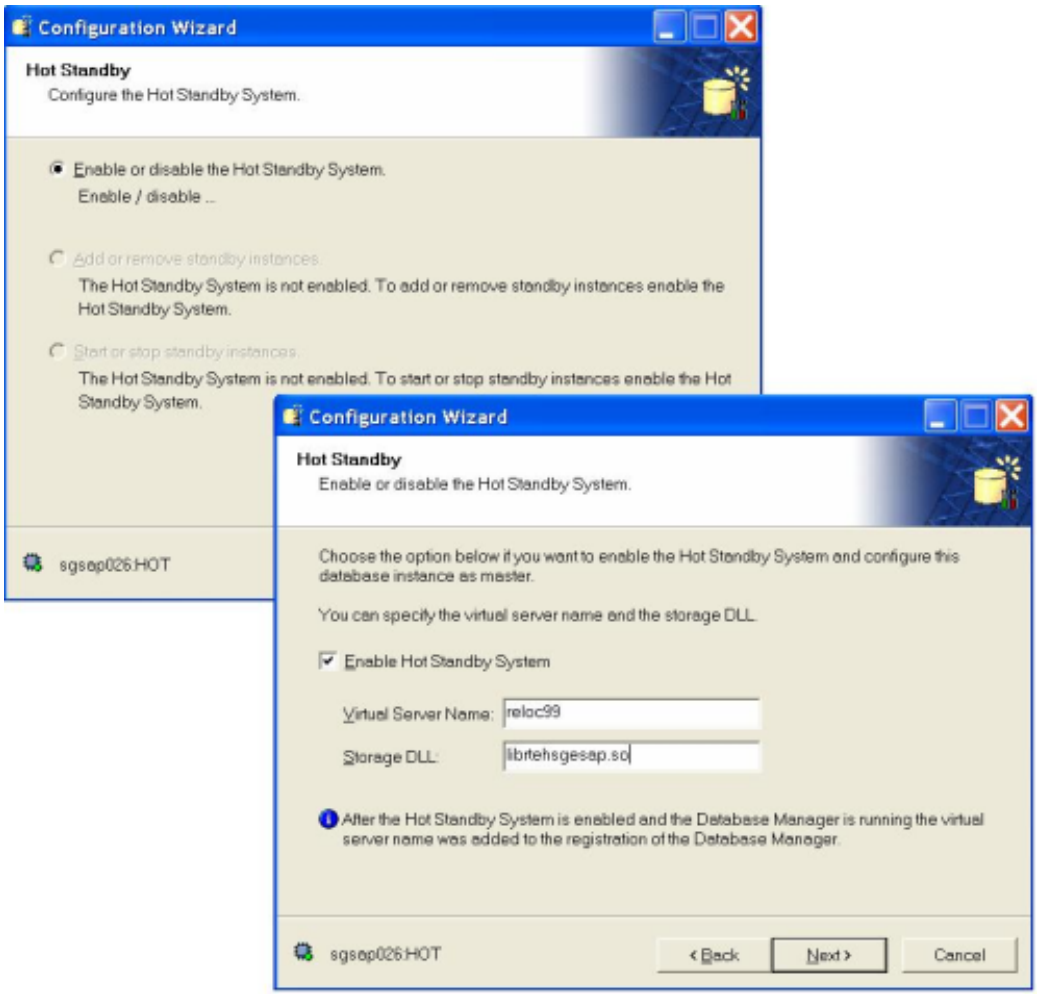
The activation of the hot standby happens on the master instance.

Logon to the instance via dbmcli and issue the following commands:

```
hss_enable node=<LCRELOC>
lib= /opt/cmcluster/sap/lib/hpux64/librtehssgesap.so
hss_addstandby <secondary_hostname> login=<osusr>,<pwd>
```

Alternatively, you can use the configuration dialog of the Hot Standby button in the SAP Database Manager configuration to set these values. see Figure 4-2.

Figure 4-2 Hot Standby System Configuration Wizard Screens



SGeSAP Modular Package Configuration

This section describes how to configure the SGeSAP `lc` package.

liveCache Installation Step: LC171

Create an ASCII package configuration file for a Livecache package:

```
cmmakepkg -m sgesap/livecache lc<LCSID>.config
```

liveCache Installation Step: LC172

Customize the content of the liveCache package configuration file.

Recommended standard Serviceguard parameter settings for customization of `<LCSID>.config` are:

```
package_name lc<LCSID>
```

```
package_type failover
:
ip_address …
:
service_name lc<LCSID>knl
service_cmd "/sapdb/<LCSID>/db/sap/lccluster monitor"
service_restart 3
```

All other standard parameters should be chosen as appropriate for the individual setup.

The sgesap/livecache module parameters of the package configuration file are as follows:

`lc_system` determines the name of the clustered liveCache.

The `lc_virtual_hostname` specified corresponds to the virtual hostname of the liveCache. For hot standby liveCache systems, this is the same value that gets specified with `hss_enable` in `dbmcli`. The virtual hostname is a string value. It is not possible to specify the corresponding ipv4 or ipv6 address instead. If the string is empty, the DNS resolution of the first specified `ip_address` will be substituted. In this case, the script only works properly if reliable address resolution is available. Domain name extensions are not part of the virtual hostname.

`lc_startmode` defines the operation mode into which the Liveache should be started automatically as part of a package startup/failover operation. Legal values are

`offline` = only vserver will be started

`admin` = Livecache started in admin mode

`slow` = Livecache started in cold -slow mode

`online` = Livecache started in online mode

`lc_userkey` sets the Livecache userkey that is mapped for the operating system level administrator to the Livecache control user (via XUSER settings). The value needs to be set if the default control userkey 'c' is not used.

liveCache Installation Step: LC172

If hss is required, additional hot standby parameters can be set.

A hot standby initialization during startup might require hardware-based copy mechanisms of the underlying storage array. `lc_copy_mechanism` defines which hardware-based copy mechanism is to be used. Currently the only supported value is `businesscopy`.

A hot standby Livecache instance does not detect a Livecache shutdown. It won't reconnect to the restarted Livecache master without external trigger. The hot standby cluster package will send this trigger, except `lc_standby_restart` is set to `no`.

A hot standby initialization procedure can perform a plausibility check of its storage configuration. Pre-saved ids of LUNs are compared with current ids to prevent that outdated configuration information is used for cloning. This adds to the robustness of the initialization process, but for large implementations it significantly slows down the standby startup. By setting `lc_avoid_idcheck` to no the check can be turned off.

liveCache Installation Step: LC211

Log in on the primary node that has the shared logical volumes mounted.

Create a symbolic link that acts as a hook that informs SAP software where to find the liveCache monitoring software to allow the prescribed interaction with it. Optionally, you can change the ownership of the link to `sdb:sdba`.

```
ln -s /etc/cmcluster/<LCSID>/saplc.mon \ /sapdb/<LCSID>/db/sap/lccluster
```

liveCache Installation Step: LC216

For the following steps the SAPGUI is required. Logon to the APO central instance as user SAP*.

Start transaction `/nlc10` and enter LCA for the logical connection.

```
/nlc10 -> Logical connection -> LCA -> liveCache -> Create/Change/Delete
Connection
```

Change liveCache server name to the virtual IP name for the liveCache and save it.

Change the liveCache instance name to <LCSID>. Redo the above steps for LDA.

SGeSAP Legacy Package Configuration

This section describes how to configure the SGeSAP lc package.

liveCache Installation Step: LC165

Install the product depot file for SGeSAP (T2803BA) using swinstall (1m) if this has not been done already.

The installation staging directory is /opt/cmcluster/sap. All original product files are copied there for reference purposes.

liveCache Installation Step: 167

Copy the relevant SGeSAP files for liveCache to the cluster directory:

```
mkdir /etc/cmcluster/<LCSID>
chown root:sapsys /etc/cmcluster/<LCSID>
chmod 775 /etc/cmcluster/<LCSID>
cp /opt/cmcluster/sap/SID/saplc.mon /etc/cmcluster/<LCSID>
cp /opt/cmcluster/sap/SID/sapwas.cntl /etc/cmcluster/<LCSID>
cp /opt/cmcluster/sap/SID/customer.functions/ /etc/cmcluster/<LCSID>
cp /opt/cmcluster/sap/SID/sap.config /etc/cmcluster/<LCSID>
```

If this has not already been done, also copy the system independent integration files:

```
cp /opt/cmcluster/sap/*.functions /etc/cmcluster
```

liveCache Installation Step: LC170

Create standard package control and configuration files:

▲

```
cd /etc/cmcluster/<LCSID>
cmmakepkg -s lc<LCSID>.control.script
cmmakepkg -p lc<LCSID>.config
```

Both files need to be adapted according to reflect the filesystem layout option chosen and the individual network setup. For general information on how to customize the package control and configuration files refer to the Managing Serviceguard manual, which is available on http://docs.hp.com/en/ha.html.

▲ For lc<LCSID>.config the following settings are strongly recommended:

```
PACKAGE_NAME lc<LCSID>
PACKAGE_TYPE FAILOVER
RUN_SCRIPT /etc/cmcluster/<LCSID>/lc<LCSID>.control.script
HALT_SCRIPT /etc/cmcluster/<LCSID>/lc<LCSID>.control.script
SERVICE_NAME SGeSAPlc<LCSID>
```

▲ Forlc<LCSID>.cntl the following settings are strongly recommended:

```
SERVICE_NAME[0]="SGeSAPlc<LCSID>"
SERVICE_CMD[0]="/sapdb/<LCSID>/db/sap/lccluster monitor"
SERVICE_RESTART[0]="-r 3"
```

▲ All other parameters should be chosen as appropriate for the individual setup.

liveCache Installation Step: LC180

In /etc/cmcluster/<LCSID>/lc<LCSID>.control.script add the following lines to the customer defined functions.

These commands will actually start and stop the liveCache instance.

```
function customer_defined_run_cmds
{
```

```
### Add following line
. /etc/cmcluster/<LCSID>/sapwas.cntl start <LCSID>
test_return 51
}
function customer_defined_halt_cmds
{
### Add following line
. /etc/cmcluster/<LCSID>/sapwas.cntl stop <LCSID>
test_return 52
}
```

liveCache Installation Step: LC190

The /etc/cmcluster/<LCSID>/sap.config configuration file of a liveCache package is similar to the configuration file of Netweaver Instance packages.

The following standard parameters in sap.config have to be set for a liveCache package:

```
LCRELOC=<reloclc_s>
TRANSRELOC=<relocsapnfs_s>
CLEANUP_POLICY= [ strict | normal | lazy ]
```

The strict option can be set in order to guarantee the liveCache package gets all resources required on the adoptive node. Remember that this option can crash a system running on the adoptive node.

▲ The following liveCache specific parameters in `sap.config` have to be set:

`LCSTARTMODE` specifies the mode of the liveCache database up to which it should be started automatically. Possible values for version 7.4 of liveCache are:

`OFFLINE`: only vserver will be started

`ADMIN`: liveCache started in cold mode

`SLOW`: liveCache started in cold -slow mode

ONLINE: liveCache started in warm mode

In liveCache version 7.5 some of these values have changed: (<from> -> <to>)

```
COLD -> ADMIN
WARM -> ONLINE
```

It is recommended to use the new values for liveCache version 7.5. For compatibility reasons the original values will still work though.

After failover, liveCache will only be recovered to the state referenced in `LCSTARTMODE`.

The behavior of the liveCache service monitoring script is independent from the setting of `LCSTARTMODE`. The monitoring script will always monitor the vserver process. It will start to monitor if liveCache is in `WARM` state, as soon as it reaches `WARM` state for the first time.

The `LCMONITORINTERVAL` variable specifies how often the monitoring polling occurs (in sec.)

hot standby Installation Step: LC193

The following additional parameters in sap.config have to be set for a hot standby liveCache package:

Hot standby systems have a preferred default machine for the instance with the master role. This should usually correspond to the hostname of the primary node of the package. The hostname of the failover node defines the default secondary role, i.e. the default standby and alternative master machine. For a hot standby system to become activated, the LC_PRIMARY_ROLE and the LC_SECONDARY_ROLE need to be defined with physical hostnames. No virtual addresses can be used here.

```
LC_PRIMARY_ROLE=<primary_node>
LC_SECONDARY_ROLE=<secondary_node>
```

A hot standby initialization during startup might require hardware-based copy mechanisms of the underlying storage array. Currently HP StorageWorks XP arrays with BCV volumes are supported. The following parameter needs to be set:

```
LC_COPY_MECHANISM=BUSINESSCOPY
```

Optional hot standby Installation Step: LC197

A hot standby initialization procedure can perform a plausibility check of its storage configuration.

A hot standby initialization procedure can perform a plausibility check of its storage configuration. Presaved world-wide IDs of LUNs are compared with current world-wide IDs to prevent that outdated configuration information is used for cloning. This adds to the robustness of the initialization process, but for large implementations it significantly slows down the standby startup. By setting:

```
AVOID_WWID_CHECK=1
```

in `sap.config` the check can be turned off.

Optional hot standby Installation Step: LC198

The hot standby liveCache instance does not detect a regular liveCache master shutdown.

It will not reconnect to the restarted liveCache master. The `sap.config` parameter

```
LC_STANDBY_RESTART=1
```

will (re)start a remote hot standby system after the master liveCache instance package is started.

Optional hot standby Installation Step: LC200

You can optionally handle SAP ABAP Application Servers with a liveCache package.

Refer to Chapter 3 - SGeSAP Configuration to get more information on the usage of the parameters `ASSID`, `ASHOST`, `ASNAME`, `ASNR`, `ASTREAT` and `ASPLATFORM`

liveCache Installation Step: LC205

Distribute the content of *`/etc/cmcluster/<LCSID>`* ***to all cluster nodes on which the liveCache should be able to run.***

liveCache Installation Step: LC210

Log in on the primary node that has the shared logical volumes mounted.

Create a symbolic link that acts as a hook that informs SAP software where to find the liveCache monitoring software to allow the prescribed interaction with it. Optionally, you can change the ownership of the link to `sdb:sdba`.

```
ln -s /etc/cmcluster/<LCSID>/saplc.mon \ /sapdb/<LCSID>/db/sap/lccluster
```

liveCache Installation Step: LC215

For the following steps the SAPGUI is required. Logon to the APO central instance as user SAP*.

Start transaction `/nlc10` and enter LCA for the logical connection.

```
/nlc10 -> Logical connection -> LCA -> liveCache -> Create/Change/Delete
Connection
```

Change liveCache server name to the virtual IP name for the liveCache and save it.

Change the liveCache instance name to `<LCSID>`. Redo the above steps for LDA.

Livecache Service Monitoring

SAP recommends the use of service monitoring in order to test the runtime availability of liveCache processes. The monitor, provided with SGeSAP, periodically checks the availability and responsiveness of the liveCache system. The sanity of the monitor will be ensured by standard Serviceguard functionality.

The liveCache monitoring program is shipped with SGeSAP in the saplc.mon file. The monitor runs as a service attached to the `lc<LCSID>` Serviceguard package.

If the monitor recognizes the liveCache to be unavailable, it will promote the hot standby system to become the new master. If no hot standby is available, the monitor will try to restart liveCache on the node it is currently running. If this does not succeed, the runtime operating system resources of liveCache are cleaned up and another local restart attempt is made. If this still does not bring liveCache back to work and a failover

package with storage option 1,2 or 3 is used, Serviceguard will switch the package and try to restart the same instance on different hardware.

Monitoring begins with package startup. At this point, the monitor will make sure, that liveCache is working only up to the point that is specified in `lc_startmode (legacy: LCSTARTMODE)`. For example, if the mode is set to offline, only the `vserver` processes will be part of the monitoring. Still, the monitor detects any manual state change of liveCache. By subsequent manual operations, liveCache will be entering cold state and finally warm state, which the monitor automatically detects. As soon as the liveCache reaches the warm state once, the monitoring increases its internal monitoring level to the same state.

As any SGeSAP package, the liveCache package will skip SAP specific startup steps when during startup a debug file `/var/adm/cmcluster/debug<packagename > (legacy: /etc/cmcluster/<LCSID>/debug)` is found. This is useful for debugging purposes to allow access to the log files placed on shared logical volumes if a package does not start up to its full extend. The liveCache monitor is started regardless of the availability of a debug file. The monitor will detect the existence of the file and will enter a pause mode until it is removed. This is valid not only during package startup, but also at runtime. As a consequence, the monitoring of a fully started package will be paused by the global debug file.

NOTE: Activation of pause mode, state changes of liveCache and liveCache restart attempts get permanently logged into the standard package logfile

The monitor can also be paused by standard administrative tasks that use the administrative tools delivered by SAP. Stopping the liveCache using the SAP `lcinit` shell command or the APO LC10 transaction will send the monitoring into pause mode. This prevents unwanted package failovers during liveCache administration. Restarting the liveCache in the same way will also trigger reactivation of the monitoring. The MAXDB Database Manager GUI is not yet cluster-aware and must not be used to stop liveCache in combination with Serviceguard.

liveCache Installation Step: LC210

Log in on the primary node that has the shared logical volumes mounted.

Create a symbolic link that acts as a hook that informs SAP software where to find the liveCache monitoring software to allow the prescribed interaction with it. Optionally, you can change the ownership of the link to `sdb:sdba`.

```
ln -s /etc/cmcluster/<LCSID>/saplc.mon \ /sapdb/<LCSID>/db/sap/lccluster
```

liveCache Installation Step: LC215

For the following steps the SAPGUI is required. Logon to the APO central instance as user SAP*.

Start transaction`/nlc10` and enter LCA for the logical connection.

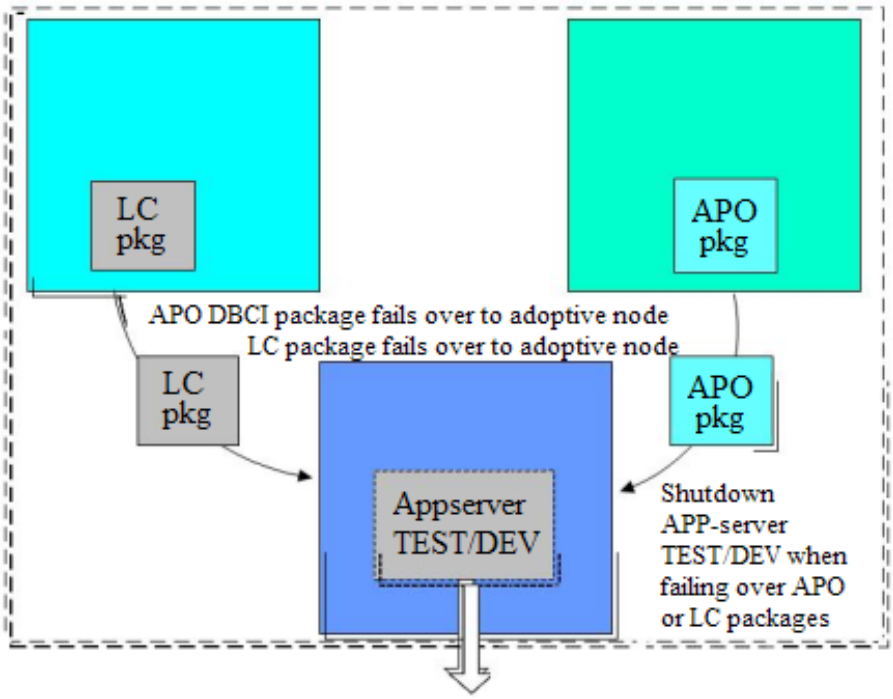
```
/nlc10 -> Logical connection -> LCA -> liveCache -> Create/Change/Delete Connection
```

Change liveCache server name to the virtual IP name for the liveCache (`<relocls_s)` and save it. Change the liveCache instance name to <LCSID>. Redo the above steps for LDA.

APO Setup Changes

Running liveCache within a Serviceguard cluster package means that the liveCache instance is now configured for the relocatable IP of the package. This configuration needs to be adopted in the APO system that connects to this liveCache. Figure 4-3 shows an example for configuring LCA.

Figure 4-3 Example HA SCM Layout



liveCache Installation Step: GS220

Run SAP transaction LC10 and configure the logical liveCache names LCA and LCD to listen to the relocatable IP of the liveCache package.

liveCache Installation Step: GS230

Do the following to continue:

1. Configure the XUSER file in the APO user home and liveCache user home directories. If an .XUSER file does not already exist, you must create it.
2. The XUSER file in the home directory of the APO administrator and the liveCache administrator keeps the connection information and grant information for a client connecting to liveCache. The XUSER content needs to be adopted to the relocatable IP the liveCache is running on.

▲ To list all mappings in the XUSER file, run the following command as `sidadm` of the APO user:

```
# dbmcli -ux SAPR3,SAP -ul
```

For liveCache 7.6 or higher:

```
su - <lcsid>adm
dbmcli -ux SAP<LCSID>,<password> -ul
```

This command produces a list of `MAXDB` user keys that may be mapped to the liveCache database schema via a local hostname. SAP created keys commonly include `c`, `w` and `DEFAULT`. If a mapping was created without specifying a key, entries of the form `<num><LCSID><hostname>`, e.g. `1LC1node1` exist. These will only work on one of the cluster hosts.

▲ To find out if a given user key mapping `<user_key>` works throughout the cluster, the relocatable address should be added to the primary host using `cmmodnet -a`.

Run the following command as APO Administrator:

```
# dbmcli -U <user_key>
```

quit exits the upcoming prompt:

```
# dbmcli on <hostname> : <LCSID>> quit
```

`<hostname>` should be relocatable. If it is not, the `XUSER` mapping has to be recreated. Example:

```
mv .XUSER.62 .XUSER.62.ORG
DEFAULT key
```

```
# dbmcli -n <reloclc_s> -d <LCSID> -u SAPRIS,SAP -uk DEFAULT -us SAPRIS,SAP
-up "SQLMODE=SAPR3; TIMEOUT=0; ISOLATION=0;"
ADMIN/ key
# dbmcli -n <reloclc_s> -d <LCSID> -u control,control -uk c -us control,control
ONLINE/ key
# dbmcli -n <reloclc_s> -d <LCSID> -u superdba,admin -uk w -us superdba,admin
LCA key
# dbmcli -n <reloclc_s> -d <LCSID> -us control,control -uk 1LCA<reloclc_s> -us
control,control
```

NOTE: Refer to the SAP documentation to learn more about the dbmcli syntax. After recreation always check the connection:

```
# dbmcli -U <user_key>
# dbmcli on <reloclc_s> : <LCSID>> quit
```

liveCache Installation Step: GS240

Distribute the XUSER mappings by distributing the file *`/home/<aposid>adm/.XUSER.<version>`* ***to all APO package nodes.***

General Serviceguard Setup Changes

Dependent from the storage option chosen, the globally available directories need to be added to different existing Serviceguard packages. The following installation steps require that the system has already been configured to use the automounter feature. If this is not the case, refer to installation steps IS730 to IS770 found in Chapter 3 of this manual.

liveCache Installation Step: GS250

For option 1:

A logical volume for `/sapdb/programs` should already exist within the NFS package part of the `dbci<APOSID>` package. It contains the MAXDB client libraries. In case of a two-package installation for APO, the logical volume is part of the NFS package part of the `db<APOSID>` package.

1. Copy the content of `/sapdb/programs` from the liveCache primary node to this logical volume. It should be a superset of the files that already exist.
2. Make sure `/sapdb/programs` exists as empty mountpoint on all hosts of the liveCache and APO packages.
3. Add the following entry to `/etc/auto.direct`. on all liveCache package nodes: `/sapdb/programs <relocdbci_s>:/sapdb/programs`.

For option 2:

1. Add a shared logical volume for `/export/sapdb/programs` to the global NFS package (`sapnfs`).
2. Copy the content of `/sapdb/programs` from the liveCache primary node to this logical volume.
3. Make sure `/sapdb/programs` exists as empty mountpoint on all hosts of the liveCache package. Also make sure `/export/sapdb/programs` exists as empty mountpoint on all hosts of the sapnfs package.
4. Add the following entry to `/etc/auto.direct`.on all hosts of the liveCache package:

```
/sapdb/programs<relocsapnfs_s>:/export/sapdb/programs
```

For option 3:

1. Add two shared logical volume for `/export/sapdb/programs` and `/export/sapdb/data` to the global NFS package (`sapnfs`).
2. Copy the content of `/sapdb/programs` and the remaining content of `/sapdb/data` from the liveCache primary node to these logical volumes.

3. Make sure `/sapdb/programs` and `/sapdb/data` exist as empty mountpoints on all hosts of the liveCache package. Also make sure `/export/sapdb/programs` and `/export/sapdb/data` exist as empty mountpoints on all hosts of the sapnfs package.
4. Add the following entries to `/etc/auto.direct`. on all hosts of the liveCache package:

```
/sapdb/programs <relocsapnfs_s>:/export/sapdb/programs
/sapdb/data <relocsapnfs_s>:/export/sapdb/data
```

For option 4:

No changes are required.

liveCache Installation Step: GS270

The last step is to reconfigure the cluster with `cmapplyconf`***(1m).***

5 SAP Master Data Management (MDM)

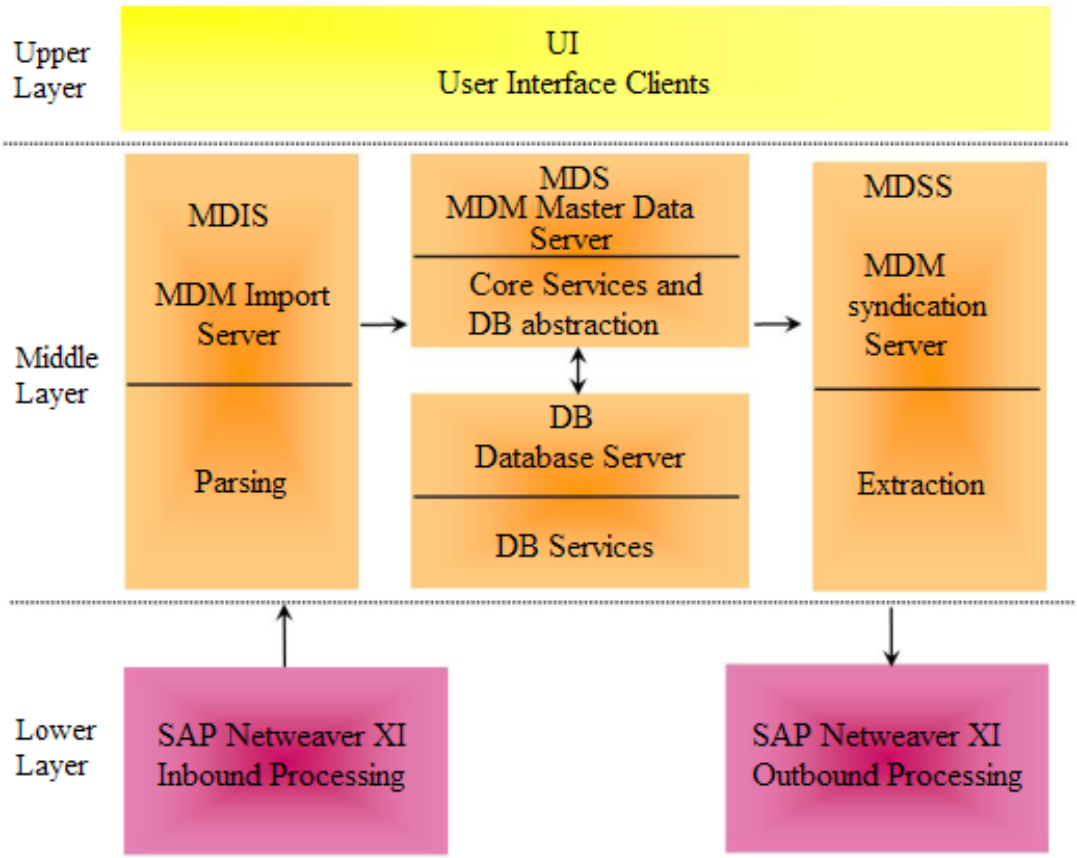
SGeSAP legacy packaging provides a cluster solution for server components of SAP **M**aster **D**ata **M**anagement (MDM) 5.5, i.e. the MDB, MDS, MDIS and MDSS servers. They can be handled as a single package or split up into four individual packages.

Master Data Management - Overview

Figure 5.1 provides a general flow of how SGeSAP works with MDM.

- The Upper Layer - contains the User Interface components.
- The Middle Layer - contains the MDM Server components. The MDM components to be clustered with SGeSAP are from the middle layer. These are MDS, MDIS, MDSS and MDB.
- The Lower Layer - contains the Inbound and Outbound connections into an SAP Netweaver XI environment.

Figure 5-1 MDM Graphical Overview



Master Data Management User Interface Components

Table 5-1 details the two MDM user interface categories, as it relates to SGeSAP:

Table 5-1 MDM User Interface and Command Line Components

Category	Description
MDM GUI (Graphical user Interface) clients	MDM Console, MDM Data Manager Client, MDM Import Manager.... Allows you to use, administer and monitor the MDM components. For example, the MDM console allows you to create and maintain the structure of MDM repositories as wellmas to control access to them. **NOTE:** The MDM GUI interface is not relevant for the implementation of the SGeSAP scripts.
MDM CLI (Command Line Interface) clients	MDM CLIX... Allows you to manage the MDM software and MDM repositories using a command line interface. The MDM CLIX interface is used by the SGeSAP scripts to manage the repositories.

MDM Server Components

The MDM Server Components provide consistent master data within a heterogeneous system landscape and allows organizations to consolidate and centrally manage their master data. The MDM Server components import and export this information to and from various clients across a network.

Table 5-2 details the MDM Server components in greater detail:

Table 5-2 MDM Server Components

Component	Description
MDS (MDM Server)	Manages one or more MDM repositories containing master data. It allows users to store, manage and update master data consisting of text, images, and other rich content, and to create taxonomies, families, and relationships. Some examples of typical MDM repositories types are: ▲ Material data ▲ Vendor ▲ Customer ▲ Employee ▲ Retail Articles ▲ Product/Material ▲ Business Partner/Vendor/Customer
MDIS (MDM Import Server)	Allows you to import data (e.g. Excel, delimited text, SQL, XML, and any ODBC-compliant source) automatically in conjunction with predefined inbound ports and import maps (e.g. restructure, cleanse, and normalize master data as part of the import process).
MDSS (MDM Syndication Server)	Allows you to export data automatically in conjunction with predefined outbound ports and syndicator maps to various remote systems (e.g. ERP master data clients, Web catalogs and files with flat or relational formats).
MDB (MDM Database Server)	Provides Database services for the MDM configuration. Currently only Oracle is supported with the combination HP-UX , SGeSAP and MDM. **NOTE:** The term MDB was chosen arbitrarily to be consistent with the other MDM names in an SGeSAP context. All these components use a leading "M".

SAP Netweaver XI components

Inbound/Outbound XI components - Provide the Inbound and Outbound plug-ins to allow the connection of the MDM system to remote systems.

NOTE: Go to `http://service.sap.com/installmdm` to access the SAP Service Marketplace web site, where you can then access the required SAP documentation needed for this installation.

You must be a SAP registered user to access these documents.

MDM 5.5 SP05 - Master Guide

MDM 5.5 SP05 - Console Reference Guide

MDM 5.5 SP05 - Release Notes

MDM 5.5 SP05 - Installation Guide

MDM 5.5 SP05 - Solution Operation Guide

As of the writing of this document, the following SAP notes contain the most recent information on the MDM installation as well as corrections to the installation:

1025897 - MDM 5.5 SP05 Release Note

822018 - MDM 5.5 Release Restriction Note

Installation and Configuration Considerations

The following sections contain a step-by-step guide on the components required to install MDM in a SGeSAP (Serviceguard extension for SAP) environment.

The MDM server components that are relevant for the installation and configuration of the SGeSAP scripts are: MDS, MDIS, MDSS and MDB.

Prerequisites

You must having the following installed and already configured:

- HP-UX and Serviceguard
- A Serviceguard cluster with at least two nodes attached to the network. (Node names: `clunode1` and `clunode2`)
- Any shared storage supported by Serviceguard. The shared storage used for this configuration is based on (**E**nterprise **V**irtual **A**rray) EVA - a fibre channel based storage solution.

NOTE: Refer to the latest Managing Serviceguard manual for Serviceguard installation instructions from `docs.hp.com | High Availability | Serviceguard`.

The MDM SGeSAP File System Layout

The following file system layout will be used for the MDM Server components.

`/oracle/MDM`

For performance reasons, the MDM database (`MDB`) file systems will be based on local and relocatable storage (the physical storage volume / file system can relocate between the cluster nodes, but only ONE node in the cluster will mount the file system). The file system will be mounted by the cluster node on which the database instance is started/running. The directory mount point is `/oracle/MDM`. All I/O is local to the cluster node.

`/home/mdmuser`

`/export/home/mdmuser`

These are the file systems for the MDM server components (These are MDS, MDSS and MDIS) that are of dynamic in nature - e.g. configuration files, log files, import files, export files). These file systems will be based on NFS. One cluster node mounts the physical storage volume and runs as a NFS server, exporting the file systems to NFS clients. Mount point for the NFS server file system is `/export/home/mdmuser`.

All nodes in the cluster mount the NFS exported files systems as NFS clients. Mount point for the NFS client file system is `/home/mdmuser`. Each of the MDM server (`mds`, `mdss` and `mdis`) components will use it's own directory (e.g.: `/home/mdmuser/mds`, `/home/mdmuser/mdss`, `/home/mdmuser/mdis`) within `/home/mdmuser`.

Should the NFS server node fail - then the storage volume will be relocated to another cluster node, which then will take over the NFS server part.

NOTE: The advantage of the NFS server/client approach from a system management viewpoint is only one copy of all the MDM server files have to be kept and maintained on the cluster instead of creating and distributing copies to each cluster node. Regardless of on which node any of the MDM server components are running in the cluster - the MDM server files are always available in the`/home/mdmuser` directory.

The disadvantage: I/O performance might become a bottleneck. Should IO performance become an issue then it would become necessary to split and to create local/relocatable file systems for `/home/mdmuser/mds`, `/home/mdmuser/mdss`,`/home/mdmuser/mdis`.

/opt/MDM

The /opt/MDM mount point is used for storing the binaries of the MDM servers. These are static files installed once during the installation. In this specific cluster configuration each cluster node had it's own copy of the /opt/MDM file system. The/opt/MDM only contains a few files and because of it's small size, a separate storage volume / mount point was not created but instead kept in the existing /opt directory. The disadvantage here is a copy has to be maintained (for each software installation) on each cluster member. From a system management an alternative approach would be to configure /opt/MDM as an NFS file system.

Single or Multiple MDM Serviceguard Package Configurations

MDM with SGeSAP can be configured to run either as a "Single" Serviceguard package (ONE package) OR as "Multiple" Serviceguard packages (FOUR+ONE).

Single MDM Serviceguard Package (ONE)

- In the case of "Single" MDM Serviceguard package all four MDM components (mdb mds mdis mdss) are always started and stopped together on "one" cluster node - the node on which the Serviceguard package itself is started or stopped on. All four MDM components are combined or grouped together. This is also is reflected in the name for this Serviceguard package name called mgroupMDM.
- The naming convention for the Serviceguard packages is: mgroupID - the string "mgroup" stands for mdm grouped components and is required; the string ID is used as a placeholder for a xx System ID and can be set to any value.
- The mgroupMDM package contains code for dependency checks: it will ensure that the MDM components are started and stopped in the correct order: mdb -> mds -> mdis -> mdss - and in the reverse order for stop.
- The advantage of the Single MDM Serviceguard package is a simpler setup - less steps are involved in setting up and running the configuration - but with the limitation that all four MDM server components always have to run together on the same cluster node.

Multiple MDM Serviceguard packages (FOUR+ONE)

- In the case of "Multiple" MDM Serviceguard packages each of the four MDM component (mdb, mds, mdis and mdss) is configured and run in a separate (mdbMDM, mdsMDM, mdssMDM, mdisMDM) Serviceguard package. So in the case of a four node cluster each MDM Serviceguard package could be configured to run on it's own cluster node and by this distribute load over the four cluster nodes.
- The naming convention for the Serviceguard packages is: mdbMDM - the string mdb stands for mdm database components and is required; the string MDM is used as a placeholder for a System ID and can be set to any value. The name of the other Serviceguard packages is derived from: mds stands for MDM master data server components, mdis stands for MDM import server components and mdss stands for MDM syndication server components.
- A "fifth" Serviceguard package called masterMDM must also be configured for this configuration type. It will be used to ensure that cluster-wide the four MDM Serviceguard packages are started in the correct order: mdbMDM -> mdsMDM -> mdisMDM -> mdssMDM regardless of on which cluster node they are started. The same is true for stopping the packages - the reverse order will be used.
- The advantage of Multiple MDM Serviceguard packages is flexibility: it is possible to balance the load by distributing the MDM servers over cluster nodes - but with the cost of requiring more steps to install and run the system.

Setup Step: MDM010

Add MDM virtual IP addresses and hostnames to /etc/hosts.

The database will be accessible under the 172.16.11.97/mdbreloc virtual IP address/hostname, the MDS component under the 172.16.11.95/mdsreloc virtual IP address/hostname and the MDM NFS files systems under 172.16.11.95/mdmnfsreloc virtual IP address/hostname. MDIS and MDSS do not require a separate virtual address. Both always connect to the virtual IP address of the MDS server. A virtual IP address will be enabled on the cluster node where the corresponding Serviceguard package is started and disabled when the package is stopped.

```
/etc/hosts
----------
172.16.11.95 mdsreloc # MDM reloc address for MDS
172.16.11.96 mdmnfsreloc # MDM reloc address for NFS
172.16.11.97 mdbreloc # MDM reloc address for DB
172.16.11.98 clunode1 # cluster node 1
172.16.11.99 clunode2 # cluster node 2
```

Setup Step: MDM020

Run `ioscan` and `insf` to probe/install new disk devices.

Scan for new disk devices on the first cluster node (`clunode1`). In the following example, the storage used for this configuration is EVA / HSV200 based fibre storage and was already initialized.

```
iocan -fnC disk
insf -e
disk 175 0/2/0/0.1.10.0.0.0.2 sdisk CLAIMED DEVICE HP HSV200 /dev/dsk/c12t0d2
/dev/rdsk/c12t0d2
disk 178 0/2/0/0.1.10.0.0.0.5 sdisk CLAIMED DEVICE HP HSV200 /dev/dsk/c12t0d5
/dev/rdsk/c12t0d5
```

Setup Step: MDM030

Create file systems and mount points on the first cluster node.

The following commands create the LVM file systems and mount points for the MDM components. The database file system in this configuration has a size of 20 GB (`/oracle/MDM`), the file system for the MDM components (`/export/home/mdmuser`) has a size of 7GB.

NOTE: The file system`/export/home/mdmuser` will be used by all 3 MDM server components MDS, MDIS and MDSS.

```
clunode1:
mkdir /dev/vgmdmoradb
mkdir /dev/vgmdmuser
mknod /dev/vgmdmoradb/group c 64 0x500000
mknod /dev/vgmdmuser/group c 64 0x530000
pvcreate /dev/rdsk/c12t0d2
pvcreate /dev/rdsk/c12t0d5
vgcreate /dev/vgmdmoradb /dev/dsk/c12t0d2
vgcreate /dev/vgmdmuser /dev/dsk/c12t0d5
lvcreate -L 20000 -n lvmdmoradb /dev/vgmdmoradb
lvcreate -L 7000 -n lvmdmuser /dev/vgmdmuser
newfs -F vxfs -o largefiles /dev/vgmdmoradb/rlvmdmoradb
newfs -F vxfs -o largefiles /dev/vgmdmuser/rlvmdmuser
```

Create the mount point for Oracle.

```
mkdir -p /oracle/MDM
```

Create NFS export/import mount points for the `/home/mdmuser` home directory. The directory `/export/home/mdmuser` is the NFS server (NFS export) mount point, `/home/mdmuser` is the NFS client (NFS import) mount point.

```
mkdir -p /export/home/mdmuser
mkdir -p /home/mdmuser
```

Create the mount point for the MDM binaries directory.

```
mkdir -p /opt/MDM
```

Setup Step: MDM040

Create file systems and mount points on the other cluster nodes.

All nodes of the cluster require a copy of the configuration changes. Copy LVM volume information to second cluster node:

```
clunode1:
vgchange -a n /dev/vgmdmoradb
vgexport -p -s -m /tmp/vgmdmoradb.map /dev/vgmdmoradb
vgchange -a n /dev/vgmdmuser
vgexport -p -s -m /tmp/vgmdmuser.map /dev/vgmdmuser
scp /tmp/vgmdmoradb.map clunode2:/tmp/vgmdmoradb.map
scp /tmp/vgmdmuser.map clunode2:/tmp/vgmdmuser.map
clunode2:
mkdir /dev/vgmdmoradb
mkdir /dev/vgmdmuser
mknod /dev/vgmdmoradb/group c 64 0x500000
mknod /dev/vgmdmuser/group c 64 0x530000
vgimport -s -m /tmp/vgmdmoradb.map /dev/vgmdmoradb
vgimport -s -m /tmp/vgmdmuser.map /dev/vgmdmuser
vgchange -a y /dev/vgmdmoradb
vgchange -a y /dev/vgmdmuser
vgcfgbackup /dev/vgmdmoradb
vgcfgbackup /dev/vgmdmuser
vgchange -a n /dev/vgmdmoradb
vgchange -a n /dev/vgmdmuser
mkdir -p /oracle/MDM
mkdir -p /export/home/mdmuser
mkdir -p /home/mdmuser
```

Setup Step: MDM050

Create `oramdm` **account for the Oracle user**

The following parameters and settings are used:

```
/etc/passwd
— — — —
user:oramdm
uid:205
home:/oracle/MDM
group:dba
shell:/bin/sh
/etc/group
— — — —
oper::202:oramdm
dba::201:oramdm
/oracle/MDM/.profile
— — — —
```

```
export ORACLE_HOME=/oracle/MDM/920_64
export ORA_NLS=/oracle/MDM/920_64/ocommon/NLS_723/admin/data
ORA_NLS32=/oracle/MDM/920_64/ocommon/NLS_733/admin/data export ORA_NLS32
export ORA_NLS33=/oracle/MDM/920_64/ocommon/nls/admin/data
export ORACLE_SID=MDM
NLSPATH=/opt/ansic/lib/nls/msg/%L/%N.cat:/opt/ansic/lib/\
nls/msg/C/%N.cat export NLSPATH
PATH=/oracle/MDM/920_64/bin:.:/oracle/MDM:${PATH}:/sbin
export PATH
SHLIB_PATH=/oracle/MDM/920_64/lib64:${SHLIB_PATH}:/oracle/client/92x_64/lib
export SHLIB_PATH
stty erase "^?"
set -o emacs # use emacs commands
alias __A=`echo "\020"` # up arrow = back a command
alias __B=`echo "\016"` # down arrow = down a command
alias __C=`echo "\006"` # right arrow = forward a character
alias __D=`echo "\002"` # left arrow = back a character
alias __H=`echo "\001"` # home = start of line
```

Setup Step: MDM060

Create mdm **account for the MDM user The following parameters and settings were used:**

```
/etc/passwd
— — — —
user:mdmuser
uid:205
home:/home/mdmuser
group:dba
shell:/bin/sh
/etc/group
— — — —
dba::201:oramdm,mdmuser
/home/mdmuser/.profile
— — — —
# Note: HOME=/home/mdmuser
export MDM_HOME=/opt/MDM
export PATH=${PATH}:${MDM_HOME}/bin
SHLIB_PATH=${MDM_HOME}/lib:${HOME}/oracle/MDM/920_64/lib export SHLIB_PATH
PATH=${HOME}/oracle/MDM/920_64/bin:.:${HOME}/oracle/MDM:${PA TH}:/sbin
export PATH
export ORACLE_HOME=${HOME}/oracle/MDM/920_64
export ORACLE_SID=MDM
stty erase "^?"
set -o emacs # use emacs commands
alias __A=`echo "\020"` # up arrow = back a command
alias __B=`echo "\016"` # down arrow = down a command
```

```
alias __C=`echo "\006"` # right arrow = forward a character
alias __D=`echo "\002"` # left arrow = back a character
alias __H=`echo "\001"` # home = start of line
```

Setup Step: MDM070

Copy files to second cluster node.

All nodes of the cluster require a copy of the configuration changes. Copy modified files to the second cluster node.

```
clunode1:
scp /etc/passwd clunode2:/etc/passwd
scp /etc/hosts clunode2:/etc/hosts
scp /etc/group clunode2:/etc/group
```

Setup Step: MDM080

Setup the Serviceguard directory for the `mdmNFS` package

Create a directory `/etc/cmcluster/MDMNFS` which will contain the Serviceguard templates and scripts for NFS. These scripts will mount the appropriate storage volumes, enable virtual IP addresses and export the NFS file systems to the NFS clients.

These scripts will mount the appropriate storage volumes, enable virtual IP addresses and export the NFS file systems to the NFS clients.

NOTE: The template names `mdmNFS.xxxxx` should be consistent with the variable `PACKAGE_NAME mdmNFS` as specified in the next step.

```
clunode1:
mkdir /etc/cmcluster/MDMNFS
cd /etc/cmcluster/MDMNFS
cp /opt/cmcluster/nfs/* .
cmmakepkg -s /etc/cmcluster/MDMNFS/mdmNFS.control.script
cmmakepkg -p /etc/cmcluster/MDMNFS/mdmNFS.config
clunode2:
mkdir /etc/cmcluster/MDMNFS
```

Setup Step: MDM090

Edit the Serviceguard files for the mdmNFS package.

Edit the files for the MDM package files:

The file `/etc/cmcluster/MDMNFS/mdmNFS.config` contains information on the Serviceguard package name (`mdmNFS`), the cluster nodes (*) the package can run on and the scripts to execute for running and halting `(/etc/cmcluster/MDMNFS/mdmNFS.control.script)` this package.

```
clunode1:
vi /etc/cmcluster/MDMNFS/mdmNFS.config
PACKAGE_NAME mdmNFS
NODE_NAME *
RUN_SCRIPT /etc/cmcluster/MDMNFS/mdmNFS.control.script
RUN_SCRIPT_TIMEOUT NO_TIMEOUT
HALT_SCRIPT /etc/cmcluster/MDMNFS/mdmNFS.control.script
HALT_SCRIPT_TIMEOUT NO_TIMEOUT
```

The file `/etc/cmcluster/MDMNFS/mdmNFS.control.script` contains information on the volume groups (VG[0]) , the LVM logical volumes (LV[0]), the file system mount point and options (FS...) as well as the virtual IP address (IP[0]...) to be used for this package.

```
vi /etc/cmcluster/MDMNFS/mdmNFS.control.script
VG[0]="vgmdmuser"
LV[0]="/dev/vgmdmuser/lvmdmuser"; \
FS[0]="/export/home/mdmuser"; \
FS_MOUNT_OPT[0]=""; \
FS_UMOUNT_OPT[0]=""; \
FS_FSCK_OPT[0]=""; \
FS_TYPE[0]="vxfs"
IP[0]="172.16.11.96"
SUBNET[0]="172.16.11.0"
```

The file `hanfs.sh` contains NFS directories that will exported with export options. These variables are used by the command `exportfs -i` to export the file systems and the command `exportfs -u` to unexport the file systems.

```
vi /etc/cmcluster/MDMNFS/hanfs.sh
XFS[0]="-o root=clunode1:clunode2 /export/home/mdmuser"
```

All nodes of the cluster require a copy of the configuration changes. Copy the MDM NFS Serviceguard files to the second cluster node:

```
scp -rp /etc/cmcluster/MDMNFS clunode2:/etc/cmcluster/MDMNFS
```

Setup Step: MDM100

Register the Serviceguard `mdmNFS` package in the cluster and run it:

```
clunode1
cmapplyconf -P /etc/cmcluster/MDMNFS/mdmNFS.config
cmrunpkg mdmNFS
```

NOTE: The commands executed buy the `mdmNFS` package are saved in `/etc/cmcluster/MDMNFS/nfsMDM.control.script.log`

After the cmrunpkg completes the "NFS export" directory `/export/home/mdmuser` should now be mounted on the system `clunode1`.

Setup Step: MDM110

Add `/home/mdmuser` to the automounter configuration.

The NFS exported directory will be mounted as a NFS client directory. Edit the file `/etc/auto.direct` and add the `/home/mdmuser` directory.

```
vi /etc/auto.direct
/home/mdmuser -vers=3,proto=udp \16.58.246.81:/export/home/mdmuser
```

Distribute `/etc/auto.direct` to all cluster members with `scp`.

```
scp -p /etc/auto.direct clunode2:/etc/auto.direct
```

Restart nfs subsystem on all cluster members.

```
/sbin/init.d/nfs.client stop
/sbin/init.d/nfs.client start
```

Creating an initial Serviceguard package for the MDB Component

The following steps will depend if the MDM Database will be configured as "Single" or "Multiple" MDM Serviceguard package configurations as described earlier. An "initial" Serviceguard package means that initially only the storage volumes, the file systems and the virtual IP address will be configured as a Serviceguard package. The goal is just to provide a file system for the installation of the database software. At a later step the SGeSAP specific scripts will be added to the package. Depending on the goal chose either option:

- (a) = Single MDM Serviceguard package - create a mgroupMDM package.
- (b) = Multiple MDM Serviceguard packages - create a mdbMDM package

At this stage the steps in option (a) or (b) are the same. Only the package name is different - the storage volume used is the same in both cases.

Setup Step:
Level: MDM200

(a) Single MDM Serviceguard package - create a `mgroupMDM` package.

Create Serviceguard templates and edit the files for the `mgroupMDM` package. For information on the variables used in this step see the configuration step of the nfsMDM package above.

```
clunode1:
mkdir /etc/cmcluster/MDM
cmmakepkg -s /etc/cmcluster/MDM/mgroupMDM.control.script
cmmakepkg -p /etc/cmcluster/MDM/mgroupMDM.config
vi /etc/cmcluster/MDM/mgroupMDM.config
PACKAGE_NAME mgroupMDM
NODE_NAME *
RUN_SCRIPT /etc/cmcluster/MDM/mgroupMDM.control.script
RUN_SCRIPT_TIMEOUT NO_TIMEOUT
HALT_SCRIPT /etc/cmcluster/MDM/mgroupMDM.control.script
HALT_SCRIPT_TIMEOUT NO_TIMEOUT
vi /etc/cmcluster/MDM/mgroupMDM.control.script
IP[0]="172.16.11.97"
SUBNET[0]="172.16.11.0"
VG[0]="vgmdmoradb"
LV[0]="/dev/vgmdmoradb/lvmdmoradb"; \
FS[0]="/oracle/MDM"; \
FS_MOUNT_OPT[0]=""; \
FS_UMOUNT_OPT[0]=""; \
FS_FSCK_OPT[0]=""; \
FS_TYPE[0]="vxfs"
scp -pr /etc/cmcluster/MDM clunode2:/etc/cmcluster/MDM
cmapplyconf -P /etc/cmcluster/MDM/mgroupMDM.config
cmrunpkg mgroupMDM
```

Setup Step: MDM202

(b) Multiple MDM Serviceguard packages - create a `mdbMDM` package

Create Serviceguard templates and edit the files for the mdbMDM package. For information on the variables used in this step see the configuration step of the nfsMDM package above.

```
clunode1:
mkdir /etc/cmcluster/MDM
cmmakepkg -s /etc/cmcluster/MDM/mdbMDM.control.script
cmmakepkg -p /etc/cmcluster/MDM/mdbMDM.config
vi /etc/cmcluster/MDM/mdbMDM.config
PACKAGE_NAME mdbMDM
NODE_NAME *
RUN_SCRIPT /etc/cmcluster/MDM/mdbMDM.control.script
```

```
RUN_SCRIPT_TIMEOUT NO_TIMEOUT
HALT_SCRIPT /etc/cmcluster/MDM/mdbMDM.control.script
HALT_SCRIPT_TIMEOUT NO_TIMEOUT
vi /etc/cmcluster/MDM/mdbMDM.control.script
IP[0]="172.16.11.97"
SUBNET[0]="172.16.11.0"
VG[0]="vgmdmoradb"
LV[0]="/dev/vgmdmoradb/lvmdmoradb"; \
FS[0]="/oracle/MDM"; \
FS_MOUNT_OPT[0]=""; \
FS_UMOUNT_OPT[0]=""; \
FS_FSCK_OPT[0]=""; \
FS_TYPE[0]="vxfs"
scp –pr /etc/cmcluster/MDM clunode2:/etc/cmcluster/MDM
cmapplyconf -P /etc/cmcluster/MDM/mdbMDM.config
cmrunpkg mdbMDM
```

Setup Step: MDM204

Create Serviceguard templates and edit the files for the `mdsMDM` package.

For information on the variables used in this step see the configuration step of the `nfsMDM` package above.

NOTE: As the `/home/mdmuser/mds` file system is NFS based - storage volumes or file systems have do not have to be specified in the package configuration files.

```
clunode1:
cmmakepkg -s /etc/cmcluster/MDM/mdsMDM.control.script
cmmakepkg -p /etc/cmcluster/MDM/mdsMDM.config
vi /etc/cmcluster/MDM/mdsMDM.config
PACKAGE_NAME mdsMDM
NODE_NAME *
RUN_SCRIPT /etc/cmcluster/MDM/mdsMDM.control.script
RUN_SCRIPT_TIMEOUT NO_TIMEOUT
HALT_SCRIPT /etc/cmcluster/MDM/mdsMDM.control.script
HALT_SCRIPT_TIMEOUT NO_TIMEOUT
vi /etc/cmcluster/MDM/mdsMDM.control.script
IP[0]="172.16.11.95"
SUBNET[0]="172.16.11.0"
scp –pr /etc/cmcluster/MDM clunode2:/etc/cmcluster/MDM
cmapplyconf -P /etc/cmcluster/MDM/mdsMDM.config
cmrunpkg mdsMDM
```

Setup Step: MDM206

(b) Multiple MDM Serviceguard packages - create a mdisMDM package.

Create Serviceguard templates and edit the files for the `mdisMDM` package. For information on the variables used in this step see the configuration step of the `nfsMDM` package above.

NOTE: As the /home/mdmuser/mdis file system is NFS based no storage volumes or file systems have to be specified in the package configuration files.

```
clunode1:
cmmakepkg -s /etc/cmcluster/MDM/mdisMDM.control.script
cmmakepkg -p /etc/cmcluster/MDM/mdisMDM.config
vi /etc/cmcluster/MDM/mdisMDM.config
PACKAGE_NAME mdisMDM
NODE_NAME *
RUN_SCRIPT /etc/cmcluster/MDM/mdisMDM.control.script
RUN_SCRIPT_TIMEOUT NO_TIMEOUT
HALT_SCRIPT /etc/cmcluster/MDM/mdisMDM.control.script
HALT_SCRIPT_TIMEOUT NO_TIMEOUT
```

NOTE: At this stage file /etc/cmcluster/MDM/mdisMDM.control.script does not have to be edited as neither an IP address nor a storage volume has to be configured.

```
scp –pr /etc/cmcluster/MDM clunode2:/etc/cmcluster/MDM
cmapplyconf -P /etc/cmcluster/MDM/mdisMDM.config
cmrunpkg mdisMDM
```

Setup Step: MDM208

(b) Multiple MDM Serviceguard packages - create a mdssMDM package.

Create Serviceguard templates and edit the files for the mdssMDM package. For information on the variables used in this step see the configuration step of the nfsMDM package above.

NOTE: As the /home/mdmuser/mdis file system is NFS based no storage volumes or file systems have to be specified in the package configuration files.

```
clunode1:
cmmakepkg -s /etc/cmcluster/MDM/mdssMDM.control.script
cmmakepkg -p /etc/cmcluster/MDM/mdssMDM.config
vi /etc/cmcluster/MDM/mdssMDM.config
PACKAGE_NAME mdssMDM
NODE_NAME *
RUN_SCRIPT /etc/cmcluster/MDM/mdssMDM.control.script
RUN_SCRIPT_TIMEOUT NO_TIMEOUT
HALT_SCRIPT /etc/cmcluster/MDM/mdssMDM.control.script
HALT_SCRIPT_TIMEOUT NO_TIMEOUT
```

NOTE: At this stage file /etc/cmcluster/MDM/mdssMDM.control.script does not have to be edited as neither an IP address nor a storage volume has to be configured.

```
scp –pr /etc/cmcluster/MDM clunode2:/etc/cmcluster/MDM
cmapplyconf -P /etc/cmcluster/MDM/mdssMDM.config
cmrunpkg mdssMDM
```

Setup Step: MDM210

(b) Multiple MDM Serviceguard packages - create a masterMDM package.

Create Serviceguard templates and edit the files for the masterMDM package. For information on the variables used in this step see the configuration step of the nfsMDM package above.

```
clunode1:
cmmakepkg -s /etc/cmcluster/MDM/masterMDM.control.script
cmmakepkg -p /etc/cmcluster/MDM/masterMDM.config
vi /etc/cmcluster/MDM/masterMDM.config
PACKAGE_NAME masterMDM
NODE_NAME *
RUN_SCRIPT /etc/cmcluster/MDM/masterMDM.control.script
RUN_SCRIPT_TIMEOUT NO_TIMEOUT
HALT_SCRIPT /etc/cmcluster/MDM/masterMDM.control.script
HALT_SCRIPT_TIMEOUT NO_TIMEOUT
```

NOTE: At this stage file `/etc/cmcluster/MDM/masterMDM.control.script` does not have to be edited as neither an IP address nor a storage volume has to be configured.

```
scp –pr /etc/cmcluster/MDM clunode2:/etc/cmcluster/MDM
cmapplyconf -P /etc/cmcluster/MDM/masterMDM.config
cmrunpkg masterMDM
```

Setup Step: MDM212

Start the "Oracle Server" 9.2 installation

The Oracle installer requires an X -display for output. For a remote installation start an X-Session with the following command:

```
ssh -X clunode1 -l oramdm
```

NOTE: The Oracle installation is based on version 9.2.0.0 of Oracle.

For this installation, the Database SID was set to "MDM". Replace the "MDM" string with one that is applicable to your environment.

Start the installation executing the `runInstaller` command. The following contains a summary of the responses during the installation.

```
/KITS/920_64/Disk1/runInstaller
oraInverntory
— — — —
su - /tmp/orainstRoot.sh
```

Creating Oracle Inventory pointer file (`/var/opt/oracle/oraInst.loc`)

```
INVPTR=/var/opt/oracle/oraInst.loc
INVLOC=/oracle/MDM/oraInventory
File locations
— — — —
Source:
Path products: /KITS/920_64/Disk1/products.jar
Destination:
Name: MDM (OUIHome)
Path: /oracle/MDM/920_64
Available Products
— — — —
[X] Oracle 9i Database
[ ] Oracle 9i Client
```

```
[ ] Oracle 9i Management
Installation Type
— — — —
[ ] Enterprise Edition
[X] Standard Edition
[ ] Custom
Database Configuration
— — — —
[X] General Purpose
[ ] Transaction Processing
[ ] Data Warehouse
[ ] Customized
[ ] Software Only
Database Identification
— — — —
Global Database Name: MDM
SID:MDM
Database File Location
— — — —
Directory for Database Files: /oracle/MDM/920_64/oradata
Database Character Set
— — — —
[ ] Use the default character set
[X] Use the Unicode (AL32UTF8) as the character set
Choose JDK Home Directory
— — — —
Enter JDK Home: /opt/java1.3
Summary
— — — —
-> Install
```

Setup Step: MDM214

Upgrade to Oracle 9.2.0.8

MDM requires a minimum version of 9.2.0.8 of Oracle. Before running the upgrade halt the Oracle database and halt the Oracle listener.

The 9.2.0.8 patch kit is distributed as a zip archive and must first be assembled into a Oracle install kit:

```
ssh -X hostname -l oramdm
cd /KITS/ora9208
cat p4547809_92080_HP64aa.bin p4547809_92080_HP64ab.bin > p4547809_92080.zip
/usr/local/bin/unzip p4547809_92080_HP64.zip
```

This will unzip and copy the kit into directory `/KITS/ora9208/Disk1`.

Start the Oracle upgrade executing the `runInstaller` command. The following contains a summary of the responses during the installation.

```
/KITS/ora9208/Disk1/runInstller
Specify File Locations
— — — —
```

```
Source:
Path: /KITS/oa9208/stage/products.xml
Destination:
Name:MDM
Path:/oracle/MDM/920_64
Summary
— — — —
Success
Migrate the MDM database to 9208
— — — —
```

See Oracle 9i Patch Set Notes

Release 2 (9.2.0.8) Patch Set 7 for HP-UX PA-RISC (64-bit)runInstaller

Setup Step: MDM216

Start the "Oracle Server" - Client installation for user `oramdm`

Before running the upgrade halt the Oracle database and halt the Oracle listener.

The next steps are ore installing the "Oracle Server - Client" bits to be able to run `sqlplus` command over a network connection. The "Oracle Server - Client" is installed for user `oramdm`.

Start the installation executing the `runInstaller` command. The following contains a summary of the responses during the installation.

NOTE: Oracle Server - Client version 9.2.0.8 (Based on Oracle patch set p4547809_92080.zip) is required for MDM.

```
ssh -X hostname -l oramdm
/KITS/9208/Disk1/runInstaller
File Locations
— — — —
accept defaults
Available Products
— — — —
[ ] Oracle 9i Database
[X] Oracle 9i client
[ ] Oracle 9i management
Installation Type
— — — —
[X] Administrator
[ ] Runtime
[ ] Custom
Installer Summary
— — — —
->Install
```

NOTE: SAP also provides a kit called "Oracle 9.2.0.8 client software". Do not install "Oracle 9.2.0.8 client software". The Oracle 9.2.0.8 client software is not an Oracle product. It is an SAP product and mainly provides libraries for an R/3 application server to connect to an Oracle database instance. This kit is called OCL92064.SAR and it installs into the `/oracle/client/92x_64 directory`.

Setup Step: MDM218

Start the "Oracle Server" - Client installation for user `mdmuser`

The next steps are ore installing the "Oracle Server - Client" bits so that user mdmuser will be able to run `sqlplus` commands over a network connection. The command `sqlplus system/passwd@MDM` is used to connect to the database.

The Oracle client software will be installed in directory `/home/mdmuser/oracle/MDM/920_64`.

NOTE: Oracle Server - Client version 9.2.0.8 is required for MDM.

Start the installation executing the `runInstaller` command. The following contains a summary of the responses during the installation.

```
/home/mdmuser/oracle
— — — —
mkdir -p /home/mdmuser/oracle/MDM/920_64
chown -R oramdm:dba /home/mdmuser/oracle
ssh -X hostname -l oramdm
/KITS/920_64/Disk1/runInstaller
File locations
— — — —
Source:
Path products: /KITS/Disk1/products.jar
Destination:
Name: MDM (OUIHome)
Path: /home/mdmuser/oracle/MDM/920_64
Available Products
— — — —
[ ] Oracle 9i Database
[X] Oracle 9i client
[ ] Oracle 9i management
Installation Type
— — — —
[X ] Administrator
[ ] Runtime
[ ] Custom
Choose JDK Home Directory
— — — —
Enter JDK Home: /opt/java1.3
Installer Summary
— — — —
Install
— — — —
Oracle Net Configuration Assistant
— — — —
[ ] Yes I will use a directory service
[X] No I will create net service names my self
Net Service Name Configuration
— — — —
```

```
[X] Oracle8i or late database service
[ ] Oracle 8i release 8.0 database service
NetService Name Configuration
— — — —
database used :MDM
protocol used: TCP protocol used
db host name: 172.16.11.97
Configure another service: no
Net Service Configuration Complete
Exit
End of Installation
```

Setup Step: MDM220

Upgrade to Oracle 9.2.0.8 for user `mdmuser`

Upgrade to Oracle 9.2.08 of the "Oracle Server - Client" bits with Oracle patchset p4547809_92080_HP64.zip.

The Oracle patchset has been unzipped and copied the kit into directory `/KITS/ora9208/Disk1`.

```
/KITS/ora9208/Disk1/runInstller
Specify File Locations
— — — —
Source:
Path: /KITS/oa9208/stage/products.xml
Destination:
Name:MDM
Path:/home/mdmuser/oracle/MDM/920_64
Summary
— — — —
Success
```

Setup Step: MDM222

Configure `tnsnames.ora` **and** `listener.ora`

Change into directory `/oracle/MDM/920_64/network/admin/` and edit `tnsnames.ora` and `listener.ora`. Replace the `HOST= clunode1` with the virtual address or virtual hostname for the database `172.16.11.97` or `mdbreloc`.

```
clunode1:
su - oramdm
cd /oracle/MDM/920_64/network/admin/
cp tnsnames.ora tnsnames.ora.ORG
cp listener.ora listener.ORG
/oracle/MDM/920_64/network/admin/tnsnames.ora
— — — —
# TNSNAMES.ORA Network Configuration File:
# /oracle/MDM/920_64/network/admin/tnsnames.ora
# Generated by Oracle configuration tools.
MDM =
(DESCRIPTION =
(ADDRESS_LIST =
```

```
(ADDRESS = (PROTOCOL = TCP)(HOST = 172.16.11.97)(PORT = 1521))
)
(CONNECT_DATA =
(SERVER = DEDICATED)
(SERVICE_NAME = MDM)
)
)
/oracle/MDM/920_64/network/admin/listener.ora
— — — —
# LISTENER.ORA Network Configuration File:
# /oracle/MDM/920_64/network/admin/listener.ora
# Generated by Oracle configuration tools.
LISTENER =
(DESCRIPTION_LIST =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP)(HOST = 172.16.11.97)(PORT = 1521))
)
)
)
SID_LIST_LISTENER =
(SID_LIST =
(SID_DESC =
(GLOBAL_DBNAME = MDM)
(ORACLE_HOME = /oracle/MDM/920_64)
(SID_NAME = MDM)
)
)
```

Setup Step: MDM224

Configure /`etc/tnsnames.ora`**,** `/etc/listener.ora` **and** `/etc/sqlnet.ora`

Copy the above three files in to the `/etc/directory` so the configuration information is also globally available. Copy the modified files to the second cluster node.

```
cd /oracle/MDM/920_64/network/admin/
cp tnsnames.ora /etc/tnsnames.ora
cp listener.ora /etc/listener.ora
cp sqlnet.ora /etc/sqlnet.ora
scp -p /etc/listener.ora clunode2 /etc/listener.ora
scp -p /etc/sqlnet.ora clunode2 /etc/sqlnet.ora
scp -p /etc/tnsnames.ora clunode2 /etc/tnsnames.ora
```

Start the listener and connect to the MDM database via the newly configured virtual IP address 172.16.11.97 with the following commands:

NOTE: Replace `"passwd"` with the password set during the installation.

```
su – oramdm
lsnrctl start
```

```
sqlplus system/passwd@MDM
SQL*Plus: Release 9.2.0.8.0 - Production on Fri Feb 16 04:30:22 2007
Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.
Connected to:
Oracle9i Enterprise Edition Release 9.2.0.8.0 - 64bit Production With the
Partitioning, OLAP and Oracle Data Mining options JServer Release 9.2.0.8.0 -
Production SQL>
Repeat the connect test for user mdmuser
su – mdmuser
sqlplus system/passwd@MDM
SQL*Plus: Release 9.2.0.8.0 - Production on Fri Feb 16 04:30:22 2007
Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.
Connected to: Oracle9i Enterprise Edition Release 9.2.0.8.0 - 64bit Production
With the Partitioning, OLAP and Oracle Data Mining options JServer Release
9.2.0.8.0 – Production SQL>
```

Setup Step: MDM226

Install the MDM IMPORT SERVER 5.5

NOTE: The MDM servers (`mds, mdiss, mdss`) install their files into the `/opt/MDM` directory.

Execute the following steps to install MDM Import Server:

- extract the zip archive
- uncompress and install the kit into the `/opt/MDM` directory
- create a /home/mdmuser/mdis directory
- `cd /home/mdmuser/mdis` and start the mdis process (this will create the mdis environment in the current directory)

```
su -
cd /KITS/MDM_Install/MDM55SP04P03/MDM IMPORT SERVER 5.5/HPUX_64
/usr/local/bin/unzip MDMIMS55004P_3.ZIP
cd /opt
zcat "/KITS/MDM_Install/MDM55SP04P03/MDM IMPORT SERVER\
5.5/HPUX_64/mdm-import-server-5.5.35.16-hpux-PARISC64-aCC.\ tar.Z" | tar xvf
–
MDM/bin/mdis-r, 41224248 bytes, 80517 tape blocks
MDM/bin/mdis, 355 bytes, 1 tape blocks
MDM/lib/libxerces-c.sl.26 symbolic link to libxerces-c.sl.26.
su - mdmuser
mkdir /home/mdmuser/mdis
cd mdis
mdis
```

Setup Step: MDM228

Install the MDM SERVER 5.5

NOTE: The MDM servers (`mds, mdiss, mdss`) install their files into the `/opt/MDM` directory.

Execute the following steps to install MDM Server:

- extract the zip archive
- uncompress and install the kit into the `/opt/MDM` directory
- create a `/home/mdmuser/mds` directory
- start the `mds` process (this will create the mds environment in the current directory)

```
su -
cd /KITS/MDM_Install/MDM55SP04P03/MDM SERVER 5.5/HPUX_64\
/usr/local/bin/unzip MDS55004P_3.ZIP
cd /opt
zcat "/KITS/MDM_Install/MDM55SP04P03/MDM SERVER\
5.5/HPUX_64/mdm-server-5.5.35.16-hpux-PARISC64-aCC.tar.Z" |\ tar xvf –
MDM/bin/mds-r, 41224248 bytes, 80517 tape blocks
MDM/bin/mds, 355 bytes, 1 tape blocks
.
.
su - mdmuser
mkdir /home/mdmuser/mds
cd mds
mds
```

Setup Step: MDM230

Install the MDM SYNDICATION SERVER 5.5

NOTE: The MDM servers (`mds, mdiss, mdss`) install their files into the `/opt/MDM` directory.

Execute the following steps to install MDM Syndication Server:

- extract the zip archive
- uncompress and install the kit into the `/opt/MDM` directory
- create a `/home/mdmuser/mdss` directory
- cd `/home/mdmuser/mdss` and start the mdss process (this will create the mdss environment in the current directory)

```
su -
cd /KITS/MDM_Install/MDM55SP04P03/MDM SYNDICATION SERVER\ 5.5/HPUX_64
/usr/local/bin/unzip MDMSS55004P_3.ZIP
cd /opt
zcat "/KITS/MDM_Install/MDM55SP04P03/MDM SYNDICATION SERVER\
5.5/HPUX_64/mdm-syndication-server-5.5.35.16-hpux-PARISC64\-aCC.tar.Z" |tar
xvf -
MDM/bin/mdss-r, 37537864 bytes, 73317 tape blocks
MDM/bin/mdss, 355 bytes, 1 tape blocks
.
.
su - mdmuser
mkdir /home/mdmuser/mdss
cd mdss
mdss
```

Setup Step: MDM232

Edit the `mds.ini, mdis.ini` **and** `mdss.ini` **files**

Edit the `mds.ini, mdis.ini` and `mdss.ini` files and add the virtual IP hostname for the `mds` component to the [GLOBAL] section. The `mdis` and `mdss` components use this information to connect to the `mds` server.

```
/home/mdmuser/mds/mds.ini
[GLOBAL]
Server=mdsreloc
.
.
/home/mdmuser/mdis/mdis.ini
[GLOBAL]
Server=mdsreloc
.
.
/home/mdmuser/mdss/mdss.ini
[GLOBAL]
Server=mdsreloc
.
.
```

Setup Step: MDM234

Stop all MDM processes

Before continuing with the configuration of the MDM serviceguard packages, the MDM processes have to be stopped. Use the following kill command to accomplish this:.

```
kill -9 `ps -ef|grep mdmuser|grep "\-r"|awk '{print $2}'
```

Setup Step: MDM236

(a) Single Serviceguard package - continue with `mgroupMDM` **configuration**

If the Single Serviceguard package option was chosen in the beginning of this chapter, continue with the configuration of this package. Insert in the function `customer_defined_run_cmds` and `customer_defined_halt_cmds` the following shell scripts. These scripts are responsible for executing the SGeSAP specific scripts e.g. for running and halting the MDM server components of an `mgroupMDM` package. The key statements for this package are in the beginning:

- MDM_SCR is the script to execute
- MDM_ACTION is start for running a package and stop for halting a package
- MDM_COMPONENT which of MDM components should be started or stopped, in this case the `mgroup` component

```
vi /etc/cmcluster/MDM/mgroupMDM.control.script
function customer_defined_run_cmds
{
typeset MDM_SCR="/etc/cmcluster/MDM/sapmdm.sh"
typeset MDM_ACTION="start"
typeset MDM_COMPONENT="mgroup"
${MDM_SCR} "${MDM_ACTION}" "${MDM_COMPONENT}"
test_return 51
}
function customer_defined_halt_cmds
{
```

```
typeset MDM_SCR="/etc/cmcluster/MDM/sapmdm.sh"
typeset MDM_ACTION="stop"
typeset MDM_COMPONENT="mgroup"
${MDM_SCR} "${MDM_ACTION}" "${MDM_COMPONENT}"
test_return 52
}
```

Setup Step: MDM238

(b) Multiple Serviceguard package - continue with `mdsMDM, mdisMDM, mdsMDM` **and** `masterMDM` **configuration**

If the Multiple Serviceguard package option was chosen in the beginning of this chapter, continue with the configuration of the `mdsMDM, mdisMDM, mdssMDM` and `masterMDM` packages. Insert in the function `customer_defined_run_cmds` and `customer_defined_halt_cmds` the following shell scripts. These scripts are responsible for executing the SGeSAP specific scripts e.g. for running and halting the MDM server components. The key statements for this package are in the beginning:

- MDM_SCR is the script to execute
- MDM_ACTION is start for running a package and stop for halting a package
- MDM_COMPONENT which of MDM components should be started or stopped, in this case they are mds or mdis or mdss or the master component

This code logs extensive status on the script and functions being executed.The following documents the changes for the `mdsMDM` package - only the variable `MDM_COMPONENT=mds` is changed for the `customer_defined_run_cmds`and the `customer_defined_halt_cmds` functions.

Repeat the steps below for the other packages`mdisMDM, mdssMDM` and `masterMDM`.

```
vi /etc/cmcluster/MDM/mdsMDM.control.script
function customer_defined_run_cmds
{
typeset MDM_SCR="/etc/cmcluster/MDM/sapmdm.sh"
typeset MDM_ACTION="start"
typeset MDM_COMPONENT="mds"
${MDM_SCR} "${MDM_ACTION}" "${MDM_COMPONENT}"
test_return 51
}
function customer_defined_halt_cmds
{
typeset MDM_SCR="/etc/cmcluster/MDM/sapmdm.sh"
typeset MDM_ACTION="stop"
typeset MDM_COMPONENT="mds"
${MDM_SCR} "${MDM_ACTION}" "${MDM_COMPONENT}"
test_return 52
}
```

Setup Step: MDM240

Configure `sap.config`

File `/etc/cmcluster/MDM/sap.config` contains the SGeSAP specific parameters for `MDM`.

The following table is an excerpt and explains the parameters that can be set for the `MDM` configuration.

Table 5-3 MDM parameter descriptions

Parameter	Description
MDM_DB=ORACLE	The database type being used.
MDM_DB_SID=MDM	The Oracle SID of the MDM database.
MDM_DB_ADMIN=oramdm	The Oracle admin user.
MDM_LISTENER_NAME="LISTENER"	The name of the Oracle listener.
MDM_USER=mdmuser	The MDM admin user.
MDM_MDS_RELOC=172.16.11.96	The virtual/relocatable IP address of the MDS component. The IP address is required to run clix commands. Note: MDSS and MDIS do not require a virtual IP address.
MDM_PASSWORD=""	The password used to access MDM.
MDM_REPOSITORY_SPEC=" PRODUCT_HA:MDM:o:mdm:sap \ PRODUCT_HA_INI:MDM:o:mdm:sap \"	The following a brief description of the values: Repository= PRODUCT_HADBMS instance= MDMDatabase type= o (Oracle)Username= mdmPassword= sap
MDM_CRED="Admin: "	Clix Repository Credentials: the password used for repository related commands. Note: the space in the string is required.
MDM_MONITOR_INTERVAL=60	How often the check/monitor gets executed.

MDM can be configured to run as "SINGLE" or "ONE" Serviceguard package OR as "MULTIPLE" or "FOUR+ONE" Serviceguard packages.

In the case of a "SINGLE" package all MDM components (mdb mds mdis mdss) are always started and stopped on the same cluster node which the Serviceguard package itself is started on. In this case variable MDM_MGROUP_DEPEND will be used by the Serviceguard package called (mgroupMDM) to start these in order.In the case of "MULTIPLE" Serviceguard packages each MDM component (mdb or mds or mdis or mdss) is configured as a separate Serviceguard package (mdbMDM mdsMDM mdisMDM mdssMDM) and can be started independent of the location of the other MDM Serviceguard packages. Variable MDM_MASTER_DEPEND controls this configuration and is used by a "FIFTH" Serviceguard package called (masterMDM) to start these in order.

The following table details the MDM_MGROUPand MDM_MASTER dependencies.

Table 5-4 MDM_MGROUP and MDM_MASTER dependencies

Dependency	Description
MDM_MGROUP_DEPEND="\ mdb mds mdis mdss"	The elements (mdb mds...) of variable MDM_MGROUP_DEPEND are the names of the MDM components that are started from a single Serviceguard package. The order of the MDM components is important.
MDM_MASTER_DEPEND="\ mdbMDM mdsMDM mdisMDM\ mdssMDM"	Each element (mdbMDM mdsMDM...) of variable MDM_MASTER_DEPEND is the name of a Serviceguard package to start. The position of the element determines the start order.

Setup Step: MDM242

(a) Single Serviceguard package - configure sap.config

Edit sap.config and add the following parameters for a Single Serviceguard package configuration. This is for the mgroupMDM package. The key variable for this configuration is MDM_MGROUP_DEPEND="mdb mds mdis mdss".

```
vi /etc/cmcluster/MDM/sap.config
MDM_DB=ORACLE
MDM_DB_SID=MDM
MDM_DB_ADMIN=oramdm
MDM_LISTENER_NAME="LISTENER"
MDM_USER=mdmuser
```

```
MDM_MDS_RELOC=172.16.11.96
MDM_PASSWORD=""
MDM_REPOSITORY_SPEC="
PRODUCT_HA:MDM:o:mdm:sap \
PRODUCT_HA_INI:MDM:o:mdm:sap \
PRODUCT_HA_NOREAL:MDM:o:mdm:sap \
"
MDM_CRED="Admin: "
MDM_MONITOR_INTERVAL=60
MDM_MGROUP_DEPEND="mdb mds mdis mdss"
```

Copy file `sap.config` to the second node:

```
scp -pr /etc/cmcluster/MDM/sap.config \
clunode2:/etc/cmcluster/MDM
```

Setup Step: MDM244

(b) Multiple Serviceguard package - configure `sap.config`

Edit file sap.config and add the following parameters for a Multiple Serviceguard package configuration. This data will be used for the `mdbMDM, mdsMDM, mdisMDM, mdssMDM` and `masterMDM` package. The key variable for this configuration is `MDM_MASTER_DEPEND="mdbMDM mdsMDM mdisMDM mdssMDM"`

```
vi /etc/cmcluster/MDM/sap.config
MDM_DB=ORACLE
MDM_DB_SID=MDM
MDM_DB_ADMIN=oramdm
MDM_LISTENER_NAME="LISTENER"
MDM_USER=mdmuser
MDM_MDS_RELOC=172.16.11.96
MDM_PASSWORD=""
MDM_REPOSITORY_SPEC="
PRODUCT_HA:MDM:o:mdm:sap \
PRODUCT_HA_INI:MDM:o:mdm:sap \
PRODUCT_HA_NOREAL:MDM:o:mdm:sap \
"
MDM_CRED="Admin: "
MDM_MONITOR_INTERVAL=60
MDM_MASTER_DEPEND="mdbMDM mdsMDM mdisMDM mdssMDM"
```

Copy file sap.config to second node.

```
scp -pr /etc/cmcluster/MDM/sap.config \
clunode2:/etc/cmcluster/MDM
```

Setup Step: MDM246

(a) Enable SGeSAP monitoring for the "Single" `mgroupMDM` package

The SGeSAP scripts include check functions to monitor the health of MDM processes. The variable`MDM_MONITOR_INTERVAL=60` in file sap.config determines how often the monitor function is executed. To enable monitoring it has to be configured in the Serviceguard package configuration files. The name "`mgroupMDMmon`" will be used as the "monitor name" / `SERVICE_NAME`. Edit both Serviceguard package files.

```
vi /etc/cmcluster/MDM/mgroupMDM.config
SERVICE_NAME mgroupMDMmon
```

```
SERVICE_FAIL_FAST_ENABLED NO
SERVICE_HALT_TIMEOUT 60
vi /etc/cmcluster/MDM/mgroupMDM.control.script
SERVICE_NAME[0]="mgroupMDMmon"
SERVICE_CMD[0]="/etc/cmcluster/MDM/sapmdm.sh check mgroup"
SERVICE_RESTART[0]=""
```

To activate the changes in the configuration files run the following commands:

```
cmapplyconf -P /etc/cmcluster/MDM/mgroupMDM.config
cmrunpkg mgroupMDM
```

Setup Step: MDM248

(b) Enable SGeSAP monitoring for "Multiple" MDM packages

The SGeSAP scripts include check functions to monitor the health of MDM processes. The variable `MDM_MONITOR_INTERVAL=60` in file `sap.config` determines how often the monitor function is executed. To enable monitoring it has to be configured in the Serviceguard package configuration files. For the `mdsMDM` Serviceguard package - the name "`mdsMDMmon`" will be used as the monitor name `/SERVICE_NAME`. Edit both Serviceguard package files.

```
vi /etc/cmcluster/MDM/mdsMDM.config
SERVICE_NAME mdsMDMmon
SERVICE_FAIL_FAST_ENABLED NO
SERVICE_HALT_TIMEOUT 60
vi /etc/cmcluster/MDM/mdsMDM.control.script
SERVICE_NAME[0]="mdsMDMmon"
SERVICE_CMD[0]="/etc/cmcluster/MDM/sapmdm.sh check mds"
SERVICE_RESTART[0]=""
```

To activate the changes in the Serviceguard configuration files run the following command:

```
cmapplyconf -P /etc/cmcluster/MDM/mdsMDM.config
```

Repeat the above steps for the `mdbMDM, mdisMDM` and `mdsMDM` packages.

NOTE: The `masterMDM` Serviceguard package does not require a check function. The only goal of this package is to coordinate a cluster wide start and stop of the `MDM` server packages and therefore does not need a monitor.

Setup Step: MDM250

Synchronize the `/etc/cmcluster/MDM` contents with second node

Synchronize the contents of directory `/etc/cmcluster/MDM` with the second node using following command:

```
clunode1:
scp –pr /etc/cmcluster/MDM clunode2:/etc/cmcluster/MDM
```

Setup Step: MDM252

(a) Single - Starting and Stopping the `mgroupMDM` package

Since everything is grouped in one package the following command will start and stop the package and all it's MDM server components (`mdb, mds, mdss` and `mdis`) at the same time - on the cluster node the cmrunpkg command is executed.

```
cmrunpkg mgroupMDM
cmhaltpkg mgroupMDM
```

Setup Step: MDM254

(b) Multiple - Starting and Stopping the `masterMDM` package

The masterMDM package is responsible for starting other Serviceguard packages (`mdmDB, mdsMDM, mdisMDM` and `mdssMDM`) in a cluster in the required order as specified in file sap.config. With this information, the `masterMDM` will run the appropriate cmrunpkg commands to start the packages. Before executing cmrunpkg make sure that the MDM server packages (`mdmDB, mdsMDM, mdisMDM` and `mdssMDM`) are halted from the previous installation tasks.The command `cmhaltpkg masterMDM` will stop all MDM Serviceguard packages in the reverse order as they were started.

NOTE: Since the `masterMDM` package uses standard cmrunpkg commands to start other packages - the node on which to run the underlying Serviceguard packages on - is obtained from the variable NODE_NAME as specified in files `mdmMDM.config, mdsMDM.config, mdisMDM.config` and `mdssMDM.config`.

```
cmrunpkg masterMDM
cmhaltpkg masterMDM
```

Setup Step: MDM256

Serviceguard and MDM package log files

The log files for Serviceguard packages and the command executed can be found in the following directories:

```
nfsMDM log files:
/etc/cmcluster/MDMNFS/nfsMDM.control.script.log
mgroupMDM log files:
/etc/cmcluster/MDM/mgroupMDM.control.script.log
masterMDM log files:
/etc/cmcluster/MDM/dbMDM.control.script.log
/etc/cmcluster/MDM/masterMDM.control.script.log
/etc/cmcluster/MDM/mdisMDM.control.script.log
/etc/cmcluster/MDM/mdsMDM.control.script.log
/etc/cmcluster/MDM/mdssMDM.control.script.log
Serviceguard log files:
/var/adm/syslog/sylog.log
```

6 SGeSAP Cluster Administration

SGeSAP clusters follow characteristic hardware and software setups. An SAP application is no longer treated as though it runs on a dedicated host. It is wrapped up inside one or more Serviceguard packages and packages can be moved to any of the hosts that are inside of the Serviceguard cluster. The Serviceguard packages provide a virtualization layer that keeps the application independent of specific server hardware. The virtualization is transparent in many aspects, but in some areas special considerations apply. This affects the way a system gets administered. Topics presented in this chapter include:

- Change Management
- Mixed Clusters
- Switching SGeSAP On and Off

Change Management

Serviceguard keeps information about the cluster configuration. It especially needs to know the relocatable IP addresses and its subnets, your Volume Groups, the Logical Volumes and their mountpoints. Check with your HP consultant for information about the way Serviceguard is configured to suite your SAP system. If you touch this configuration, you may have to reconfigure your cluster.

System Level Changes

SGeSAP provides some flexibility for hardware change management. If you have to maintain the server on which an (A)SCS instance is running, this instance can temporarily be moved to the host that runs its Replicated Enqueue without interrupting ongoing work. Some users might experience a short delay in the response time for their ongoing transaction. No downtime is required for the maintenance action.

If you add new hardware and SAP software needs access to it to work properly, make sure to allow this access from any host of the cluster by appropriately planning the hardware connectivity. E.g. it is possible to increase database disk space by adding a new shared LUN from a SAN device as physical volume to the shared volume groups on the primary host on which a database runs. The changed volume group configuration has to be redistributed to all cluster nodes afterwards via `vgexport(1m)` and `vgimport(1m)`.

It is a good practice to keep a list of all directories that were identified in Chapter Two to be common directories that are kept local on each node. As a rule of thumb, files that get changed in these directories need to be manually copied to all other cluster nodes afterwards. There might be exceptions. E.g. `/home/<SID>adm` does not need to be the same on all of the hosts. In clusters that do not use CFS, it is possible to locally install additional Dialog Instances on hosts of the cluster, although it will not be part of any package. SAP startup scripts in the home directory are only needed on this dedicated host. You do not need to distribute them to other hosts.

If remote shell access is used, never delete the mutual `.rhosts` entries of the root user and `<sid>adm` on any of the nodes. Never delete the secure shell setup in case it is specified for SGeSAP.

Entries in `/etc/hosts`, `/etc/services`, `/etc/passwd` or `/etc/group` should be kept unified across all nodes.

If you use an ORACLE database, be aware that the listener configuration file of SQL*Net V2 is kept in a local copy as /etc/listener.ora by default, too.

Files in the following directories and all subdirectories are typically shared:

```
/usr/sap/<SID>/DVEBMGS<INR>

/export/usr/sap/trans (except for stand-alone J2EE)

/export/sapmnt/<SID>

/oracle/<SID> or /export/sapdb
```

Chapter Two can be referenced for a full list. These directories are only available on a host if the package they belong to is running on it. They are empty on all other nodes. Serviceguard switches the directory content to a node with the package.

All directories below `/export`have an equivalent directory whose fully qualified path comes without this prefix. These directories are managed by the automounter. NFS file systems get mounted automatically as

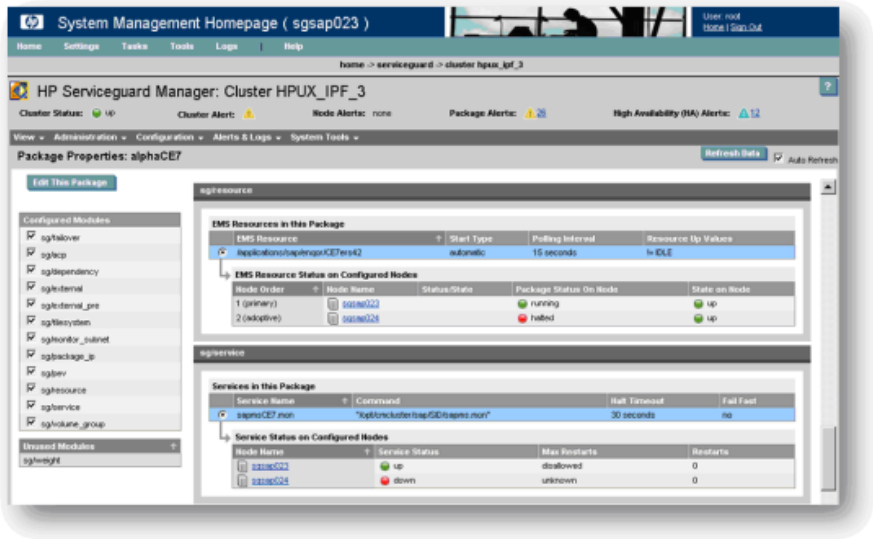
needed. Servers outside of the cluster that have External Dialog Instances installed are set up in a similar way. Refer to `/etc/auto.direct` for a full list of automounter file systems of SGeSAP.

It enhances the security of the installation if the directories below `/export` are exported without root permissions. The effect is, that the root user cannot modify these directories or their contents. With standard permissions set, the root user cannot even see the files. If changes need to be done as root, the equivalent directory below `/export` on the host the package runs on can be used as access point.

If the system is badly configured, it might be possible that the system hangs if a logon is attempted as `<sid>adm user`. The reason for this is, that `/usr/sap/<SID>/SYS/exe` is part of the path of `<sid>adm`. Without local binaries this directory links to `/sapmnt/<SID>` which in fact is handled by the automounter. The automounter cannot contact the host belonging to the relocatable address that is configured because the package is down. The system hangs. To avoid this, always keep local copies of the executables.

NOTE: If the database package with HA NFS services is halted, you cannot log in as `<sid>adm` unless you keep a local copy of the SAP binaries using `sapcpe`.

Figure 6-1 SGeSAP cluster displayed in the HP System Management Homepage



To allow proper troubleshooting there is a verbose package startup log in the package directory. It should be checked first in case of trouble. The level of information can be adjusted by changing the parameter `LOGLEVEL` in the `sap.config` file. If problems with package startup remain, a general debug mode can be activated by creating a file:

`touch /var/adm/cmcluster/debug` - debug mode for all modular SGeSAP package

`touch /var/adm/cmcluster/debug<packagename>` - debug mode for the SGeSAP modules of package `<packagename>`

`touch /etc/cmcluster/debug` - debug mode for all legacy SGeSAP packages

`touch /etc/cmcluster/<SID>/debug` - debug mode for the SGeSAP legacy SGeSAP packages of SAP system`<SID>`

The whole SGeSAP node will now be in debug mode. If the file is created in the package directory `/etc/cmcluster/<SID>`, it will only turn on the debug mode for packages in that directory.

The debug mode allows package start up to the level of SAP specific steps. All instance startup attempts will be skipped. Service monitors will be started, but they don't report failures as long as the debug mode is turned on.

In this mode it is possible to attempt manual startups of the database and/or SAP software. All rules of manual troubleshooting of SAP instances apply now. It is possible to access the application work directories to have a look at the trace files. After removal of the error cause it is vital to remove the debug file that got created:

If the package still runs, all monitors will begin to work immediately and the package failover mechanism is restored.

SAP Software Changes

During installation of the SGeSAP Integration for SAP releases with kernel <7.0, SAP profiles are changed to contain only relocatable IP-addresses for the database as well as the Central Instance. You can check this using transaction `RZ10`. In the `DEFAULT.PFL` these entries are altered:

```
SAPDBHOST= <relocatible_db_name>
rdisp/mshost= <relocatible_ci_name>
rdisp/vbname= <relocatible_ci_name>_<SID>_<INR>
rdisp/enqname= <relocatible_ci_name>_<SID>_<INR>
rdisp/btcname= <relocatible_ci_name>_<SID>_<INR>
rslg/collect_daemon/host = <relocatible_ci_name>
```

There are also two additional profile parameters `SAPLOCALHOST` and `SAPLOCALHOSTFULL` included as part of the Instance Profile of the Central Instance. Anywhere SAP uses the local hostname internally, this name is replaced by the relocatable values `<relocatable_ci_name>` or `<relocatable_ci_name>.domain.organization` of these parameters. Make sure that they are always defined and set to the correct value. This is vital for the system to function correctly.

Relocatable IP addresses can be used consistently beginning with SAP kernel 6.40. Older releases use local hostnames in profile names and startup script names. Renamed copies of the files or symbolic links exist to overcome this issue.

The destination for print formatting, which is done by a Spool Work process, uses the Application Server name. Use the relocatable name if you plan to use Spool Work processes with your Central Instance. In these cases no changes need to be done in case of a failover - printing will work persistently.

NOTE: A print job in process at the time of the failure will be canceled and needs to be reissued manually after the failover. To make a spooler highly available on the Central Instance, set the destination of the printer to `<relocatible_ci_name>_<SID>_<nr>` using the transaction `SPAD`. Print all time critical documents via the high available spool server of the Central Instance.

Print requests to other spool servers stay in the system after failure until the host is available again and the spool server has been restarted. These requests can be moved manually to other spool servers if the failed server is unavailable for a longer period of time.

Batch jobs can be scheduled to run on a particular instance. Generally speaking, it is better not to specify a destination host at all. Sticking to this rule, the batch scheduler chooses a batch server which is available at the start time of the batch job. However, if you want to specify a destination host, specify the batch server running on the highly available Central Instance. The application server name and the hostname (which is retrieved from the Message Server) are stored in the batch control tables `TBTCO`,`TBTCS`,.... In case the batch job is ready to run, the application server name is used to start it. Therefor, when using the relocatable name to build the Application Server name for the instance, you do not need to change batch jobs which are tied to it after a switchover. This is true even if the hostname which is also stored in the above tables, differs.

Plan to use `saplogon` to application server groups instead of `saptemu/sapgui` to individual application servers. When logging on to an application server group with two or more application servers, the SAP user does not need a different login procedure if one of the application servers of the group fails. Also, using login groups, provides workload balancing between application servers, too.

Within the CCMS you can define operation modes for SAP instances. An operation mode defines a resource configuration. It can be used to determine which instances are started and stopped and how the individual services are allocated for each instance in the configuration. An instance definition for a particular operation mode consists of the number and types of Work processes as well as Start and Instance Profiles. When defining an instance for an operation mode you need to enter the hostname and the system number of the application server. By using relocatable names to fill in the hostname field, the instance will be working under control of the CCMS after a failover without a change.

NOTE: If an instance is running on the standby node in normal operation and is stopped during the switchover

Only configure the update service on a node for Application Services running on the same node. As a result, the remaining servers, running on different nodes, are not affected by the outage of the update server. However, if the update server is configured to be responsible for application servers running on different nodes, any failure of the update server leads to subsequent outages at these nodes. Configure the update server on a clustered instance. Using local update servers should only be considered, if performance issues require it.

Upgrading SAP Software

Upgrading the version of the clustered SAP application does not necessarily require changes to SGeSAP. Usually SGeSAP detects the release of the application that is packaged automatically and treats it as appropriate. Make sure that you have a valid SGeSAP version in place by issuing the command:

On HP-UX 11i v2 or higher type:

```
swlist -l bundle T2357BA T2803BA
```

On HP-UX 11i type:

```
swlist -l bundle B7885BA T2803BA
```

Review the release notes of the SGeSAP version that gets reported and make sure that the target SAP application release is supported with the SGeSAP version that is already installed. If this is not the case, you should first get an update for the HP Serviceguard Extension for SAP installed and activated before continuing.

For a safe upgrade, SGeSAP can be switched off like is described in the section below.

Perform failover tests for all potential failure scenarios before putting the system in production.

If the setup has a mixed cluster of PA-RISC and IPF nodes, then he will have two sets of SAP central executables. In this case it is required to make sure that after successful upgrade on one platform, the second kernel directory gets upgraded manually. Both kernel directories need to have the same kernel revision and patch level.

Sometimes, upgrades come with additional configuration options. ASCS and Replicated Enqueue is possible beginning with ABAP kernel 4.6. SCS instances are introduced with J2EE engine 6.40. Refer to Chapter Three on how to configure packages for the additional components.

Mixed Clusters

Platform changes from HP 9000 hardware to Integrity systems are complex and require a significant investment. Changing the hardware for a multi-node cluster at once is an expensive undertaking. It is possible to perform this change step by step, replacing only one server at once with SGeSAP. During this transition phase, the cluster will technically have a mixed layout. However, from SAPs perspective it's still a homogeneous HP-UX system.

Most of SAPs business transactions are still written in ABAP, SAPs own programming language. All programs are stored as source code in the database and translated at the first time of execution if not already delivered in a translated form. At this first time of execution, the translated code is also stored in the database for next time use. This table is usually sized to hold the code for one platform. If you deploy application servers of different platforms within a single SAP system, this table needs to be sized appropriately to avoid unnecessary translations.

A mixed cluster containing both HP 9000 and Integrity HP-UX nodes must fulfill the following prerequisites:

- The system needs to be based on single-instance ORACLE database technology.
- All cluster nodes are installed with HP-UX 11i v2 (ud2) or higher.
- Serviceguard 11.16 or Serviceguard 11.17 is required
- The bundles of the HP Serviceguard Storage Management Suite are not allowed in SAP mixed clusters
- The cluster setup must follow the storage layout option 1 of chapter 2 (NFS-based)

- SGeSAP packages that do not contain database components can be configured to run on any node in the cluster. This includes `ci, jci, [a]rep` and `d` package types.
- SGeSAP database instance packages must be configured to run only on either Integrity or HP 9000 nodes. This includes `db, jdb` and `lc` package types and all package types composed of these.

NOTE: Always check for the latest versions of the Serviceguard Extension for SAP manual and release notes at `http://docs.hp.com/en/ha.html`. Especially for mixed cluster setups and configurations refer to SAP note 612796 - 'Move to HP-UX Itanium systems'.

Even though the SAP HP 9000 executables might run on Integrity systems under the ARIES binary emulator, SAP has no plans to support HP 9000 executables on Integrity systems. Consequently, it is mandatory to obtain SAP executables for the additional Integrity nodes. So the `SAPEXE` and `SAPDBEXE.SAR` archives for the Integrity based systems have to be downloaded, for example from Service Marketplace (`http://service.sap.com`) for the corresponding already installed SAP component.

The SAP kernel executables for HP 9000 and Integrity processors need to be in sync with their patch levels. It is guaranteed that the same patch levels are available from SAP at all times.

System Configuration Change: PI010

The shared disk containing SAP executables, profiles and globals (`/sapmnt/sapmnt/<SID>`) must have sufficient disk space for a full set of additional Integrity binaries.

System Configuration Change: PI020

The SAP executable directory is the key point where the redirection for HP 9000 or Integrity executables is performed, so the HP 9000 executables need to be moved to a distinct directory.

```
cd /sapmnt/<SID>
mv exe exe_pa
```

System Configuration Change: PI025

Create a directory for the Integrity binaries.

```
cd /sapmnt/<SID>
mkdir exe_ipf
```

and extract the downloaded SAR archives

```
SAR xvf <...>/SAPEXE.SAR /sapmnt/<SID>/exe_ipf
SAR xvf <...>/SAPDBEXE.SAR /sapmnt/<SID>/exe_ipf
```

System Configuration Change: PI030

The executable directory is substituted by a link referencing another local symbolic link that distinguishes between a HP 9000 or Integrity based system.

```
cd /sapmnt/<SID>
ln -s /sapmnt/<SID>exe_local exe
```

Take note that the referenced local executable link does not reside on the shared disc - it is on the local file system on each cluster node pointing to the appropriate executables for a HP 9000 or Integrity platform. In that case for a HP 9000 system:

```
cd /sapmnt
ln -s /sapmnt/<SID>/exe_pa /sapmnt/<SID>exe_local
```

System Configuration Change: PI040

Similar to the local executable link on HP 9000 systems, perform the step for the Integrity node:

```
cd /sapmnt
ln -s /sapmnt/<SID>exe_ipf <SID>exe_local
```

System Configuration Change: PI050

Make sure to have a local copy of the Oracle client libraries on all nodes.

You'll need to download the HP 9000 version for the PA RISC nodes and the IPF version for the Integrity nodes from `http://service.sap.com`.

Currently Oracle does not support the failover of databases based on HP 9000 binaries to nodes using executables compiled for the Integrity platform. Consequently this means, the use of an Oracle database package in a mixed environment is restricted to one platform. Therefore the database package must be kept either on purely HP 9000 or Integrity nodes. The installation and configuration of a database package should have been done already, as mentioned in the prerequisites. Oracle database servers at least need to match the version number that the used Oracle client libraries have. Mixed clusters are only possible using Oracle 9i versions, because the Integrity executable set of the SAP kernel uses Oracle 9i client libraries. The SAP kernel acts as an Oracle client. As a consequence, Oracle 8 database releases need to be upgraded before they can be used in mixed cluster setups. HP recommends to use the highest database server patch level that is supported for the used SAP component. This will be at least Oracle 9.2.0.3. It is not possible to use a HP 9000 based Oracle 8 database for environments that might serve SAP components on IPF.