# logo-sql08.gif

# High Availability and Disaster Recovery for Microsoft’s SAP Data Tier: A SQL Server 2008 Technical Case Study

SQL Server Technical Article

**Writers:** Joseph Sack, Eric Holling, Sanjay Mishra

**Contributors:** Juergen Thomas, Elke Bregler

**Technical Reviewers:** Prem Mehra, Danny Tambs, Cameron Gardiner, Nicholas Dritsas, Mike Weiner, Lindsey Allen, Denny Lee

**Published:** October 2010

**Applies to:** SQL Server 2008

**Summary:** Microsoft relies heavily on its SAP ERP systems to manage financial, human resources, and supply chain operations. Given the mission-critical nature of these systems, Microsoft IT has established a set of processes and technology solutions that help achieve high availability and also meet disaster recovery requirements. This technical white paper will show how Microsoft IT uses Microsoft SQL Server 2008 functionality to minimize downtime for high business impact SAP ERP applications, as well as minimizing the probability of data loss. SAP ERP uses a three-tier architecture for presentation, application, and data. This white paper will focus on the high availability techniques for SQL Server supporting the SAP data tier.

Whether your organization or company also uses SQL Server with SAP or other SQL Server driven mission critical applications, most of these concepts are applicable to any SQL Server based mission critical application and not limited to just SAP ERP systems. This document will be of interest to database administrators, IT directors, project managers, and infrastructure architects.

***Note****: This document is current as of its most recent publish date of October 28, 2010. Be sure to check www.sqlcat.com to obtain the most recent version of this paper.*

Copyright

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This white paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in, or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

© 2010 Microsoft Corporation. All rights reserved.

Contents

[Introduction 4](#_Toc275982921)

[High Availability and Disaster Recovery Solution 4](#_Toc275982922)

[Local High Availability Solution within the Primary Data Center 5](#_Toc275982923)

[Disaster Recovery Solution 5](#_Toc275982924)

[Minimizing Unplanned Downtime 6](#_Toc275982925)

[Backup and Recovery Strategy 6](#_Toc275982926)

[Annual Disaster Recovery Exercise 6](#_Toc275982927)

[Actual Disaster Recovery 6](#_Toc275982928)

[Minimizing Planned Downtime 7](#_Toc275982929)

[Security Upgrading or Updating (Operating System and Components) 7](#_Toc275982930)

[SQL Server Upgrades or Patching 8](#_Toc275982931)

[Storage Upgrades 9](#_Toc275982932)

[Server Swaps 10](#_Toc275982933)

[SAP Support Packages 11](#_Toc275982934)

[Monitoring 12](#_Toc275982935)

[Conclusion 13](#_Toc275982936)

[Appendix A: Server and Storage Specifications 13](#_Toc275982937)

[Servers 13](#_Toc275982938)

[Storage 13](#_Toc275982939)

[Appendix B: Scripts 14](#_Toc275982940)

[The sp\_MirroringStatus Procedure 14](#_Toc275982941)

# Introduction

The Microsoft IT (MSIT) SAP ERP deployment encompasses functionality around financial accounting, enterprise control, corporate finance management, treasury, real estate, project systems, benefits, payroll, sales, distribution, and material management. The application end-user base is spread across 67 countries with 4,000 named SAPGUI users and 100,000 internal and external web users. SAP ERP traffic consists of greater than 1,000 concurrent users and 1.5 million dialog steps per business day, with an average of 0.8 seconds user response time. The SAP ERP centralized database encompasses 5 terabytes of row-compressed data.

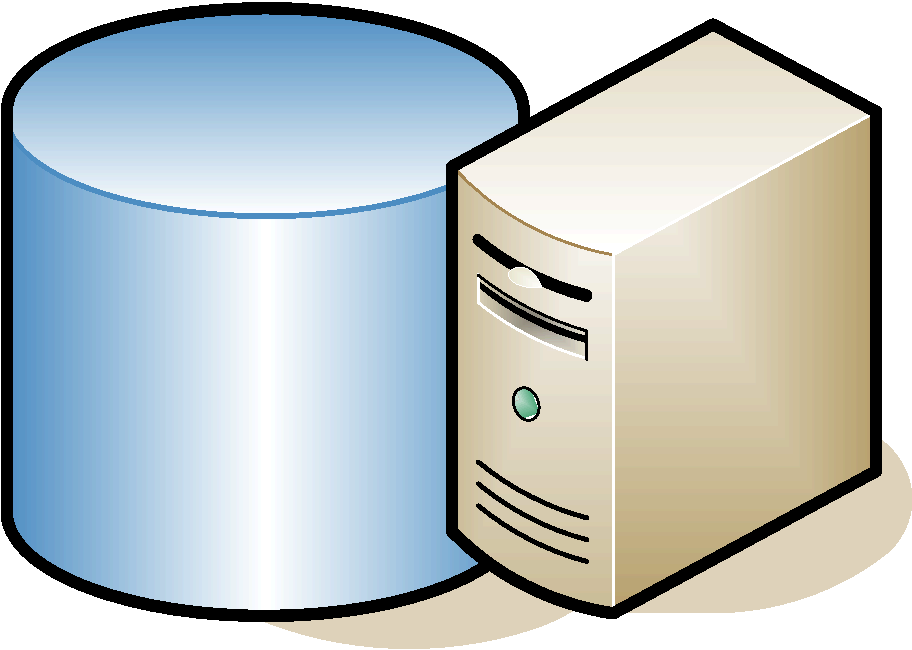
Given the high business impact, MSIT is mandated to provide the highest availability for these Tier 1 applications. Since moving to SQL Server database mirroring in SQL Server 2005 and then upgrading later to SQL Server 2008, SAP ERP production has maintained 99.995 percent uptime (includes planned security patch maintenance, as well as unplanned outages due to hardware failure; does not include planned SAP application upgrades). The high availability requirements for SAP ERP encompasses the server level, SAN storage, SAP configuration, and a fully redundant disaster recovery environment.

This white paper will describe the architecture and processes used to achieve high availability and a disaster recovery solution for Microsoft’s SAP ERP system.

# High Availability and Disaster Recovery Solution

MSIT SAP deployment uses a combination of database mirroring and log shipping to provide availability and disaster recovery solution for the data tier. Figure 1 illustrates the high availability/disaster recovery architecture deployed for MSIT SAP. For more information about server and storage specifications, see Appendix A.

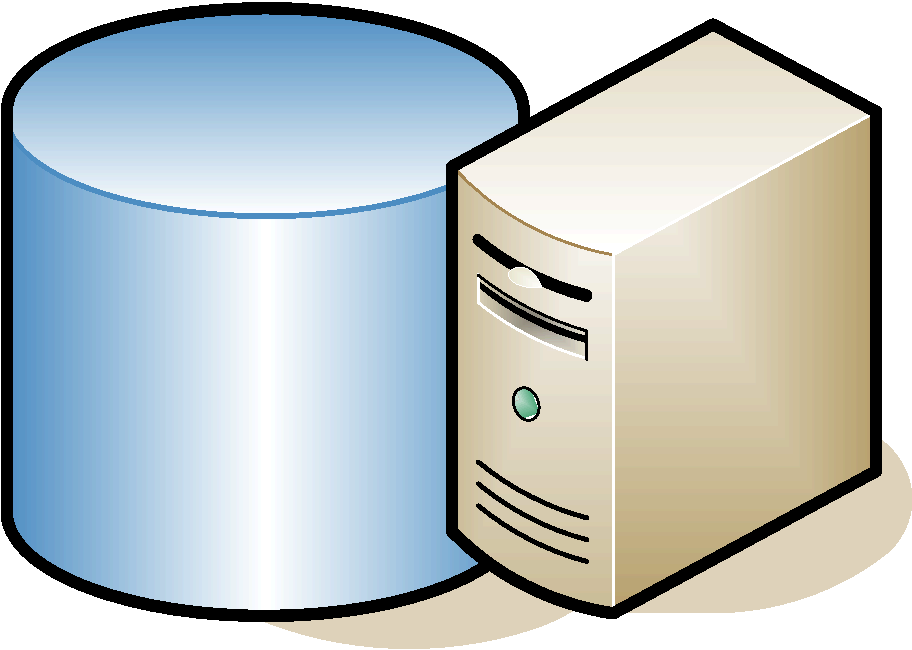
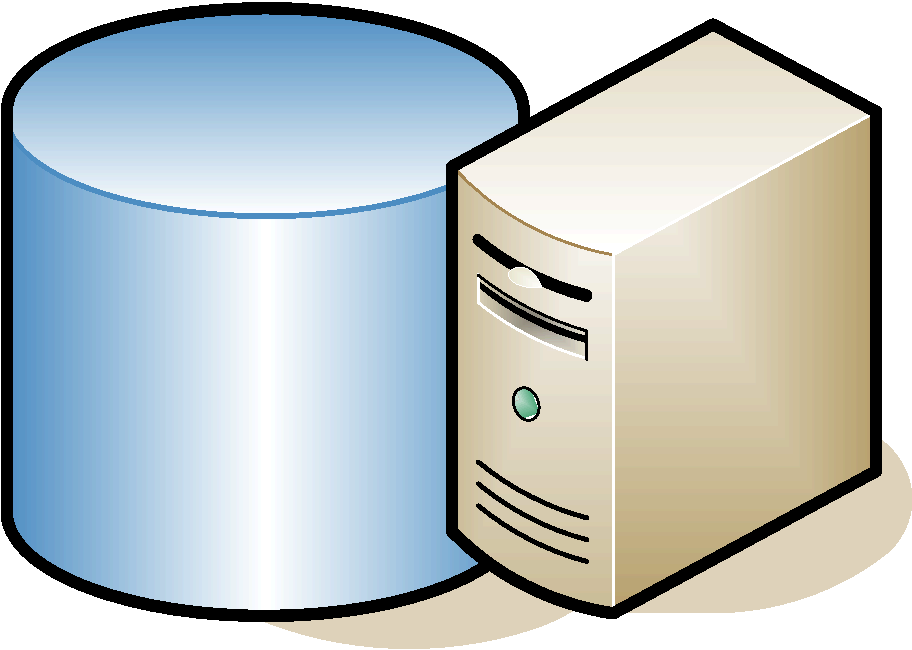
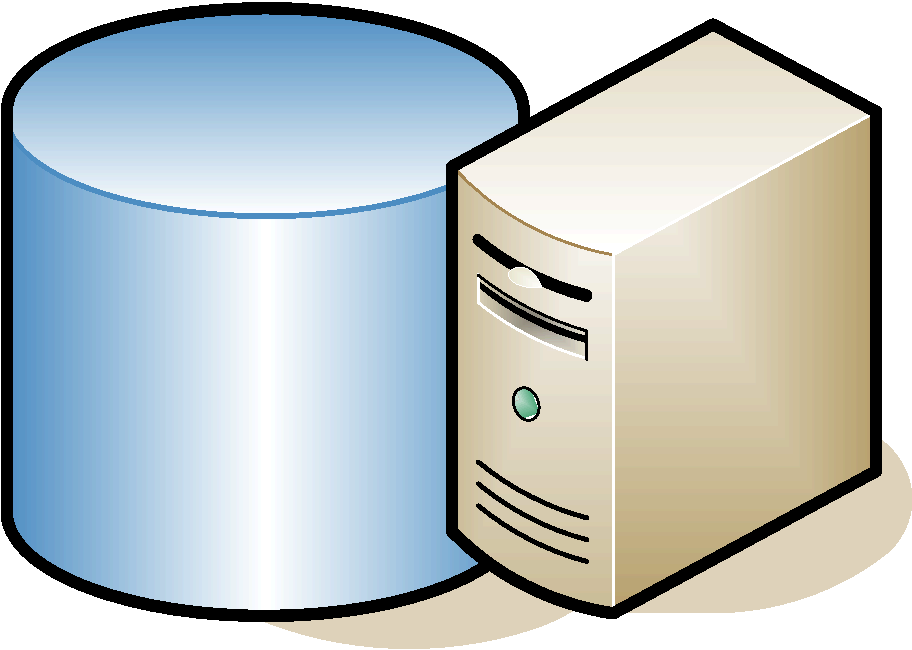
The primary data center hosts the principal, mirror, and witness servers. The disaster recovery data center, located about 160 miles away, hosts the log shipping secondary. The disaster recovery data center also hosts the test servers (not shown in the diagram). As a best practice, MSIT has deployed identical hardware and SAN configurations for production in all environments – principal, mirror, log shipping secondary, and test database servers. This is a best practice because it provides a consistent level of performance and scalability across environments. Moreover, the test environment is used for production during the disaster recovery exercise described later in this white paper.



**Primary Data Center**

**Log Shipping**

**Disaster Recovery Data Center**



**Principal**

**Mirror**

**Witness**

**Synchronous**

**Database**

**Mirroring**

**Log Shipping Secondary**

**Figure 1:** SAP ERP SQL Server High Availability and Disaster Recovery Environment

## Local High Availability Solution within the Primary Data Center

Synchronous database mirroring with a witness is used to provide database, server, and storage redundancy within the primary data center. Synchronous database mirroring provides a zero data loss solution in the event of failures of the principal server or storage components (Recovery Point Objective (RPO) = 0). The witness enables automatic failover (extremely low Recovery Time Objective (RTO)) to the mirror partner in the event of such a failure. The SAP ERP application uses the FAILOVER\_PARTNER keyword in the connection string in order for the application to be redirected to the appropriate principal. The SAP ERP application automatically reconnects work processes to SQL Server if the connection is closed or dropped due to database mirroring failover.

## Disaster Recovery Solution

To plan for the possibility of the entire primary data center becoming unavailable, MSIT chose to use log shipping to provide a remote standby database on a SQL Server instance located at a separate data center. The log shipping backup, copy, and restore jobs are scheduled to run every minute. The log backups are compressed using native SQL Server backup compression.

The log shipping has been set up to two destinations – one copy of the log backup is shipped to the disaster recovery data center, and another copy is shipped to the local mirror instance at the primary data center. On the mirror server, the restore job is disabled. This gives MSIT a redundant copy of transaction logs in the production data center, should MSIT encounter catastrophic failure on the principal.

To achieve this, MSIT has to set up log shipping in two directions between the principal and the mirror. Appropriate jobs are enabled and disabled using a custom SQL Server Agent toggle job based on the role of the particular server (principal or mirror).

# Minimizing Unplanned Downtime

MSIT SAP deployment uses the high availability/disaster recovery architecture described earlier to provide redundancy both within the primary data center and to a geographically remote disaster recovery data center. Apart from the technical solution, the MSIT team has put several procedures in place to eliminate and minimize unplanned downtime situations.

## Backup and Recovery Strategy

MSIT SAP has the following backup strategy:

* Full database backup: Weekly
* Differential backup: Daily
* Transaction log backups: Every 1 minute. Transaction log backups are done by the log shipping jobs for the disaster recovery solution.

All backup files are stored on local disk and then copied to the dry run server once a week. The dry run server is part of the restore strategy.

A backup strategy is of little use if not accompanied by a restore strategy. To verify that the backups can be successfully restored, a weekly restore is performed on a dry run server. A full backup and the subsequent differential backups are restored on the dry run server. The dry run server is located in the production data center.

Other than verifying backup/restore, this dry run server is used for running DBCC CHECKDB and verifying application changes (before changes made in production), as well as trouble-shooting and fix verification work that requires a recent copy of production data.

## Annual Disaster Recovery Exercise

To stay prepared for an actual disaster, MSIT performs an annual disaster recovery exercise. During the annual disaster recovery exercise, the servers at the primary data center are stopped, and the application and the database are brought up on the disaster recovery site. The downtime for annual disaster recovery exercise is approximately between 2 to 2 ½ hours. The SQL Server failover takes only a few minutes (because the log shipping jobs run every minute); the rest of the downtime is spent changing application configurations (this topic is not covered in this white paper). The application runs live on the disaster recovery site for a day and half, and then it is failed back to the primary data center.

## Actual Disaster Recovery

In the event of an actual disaster at the primary data center, the MSIT SAP team can quickly convert the log shipping secondary server into a production server and then redirect all application traffic to it. To ensure that the new production server has protection against failures, one of the test servers will be repurposed and set up as database mirroring partner of the new production server. Because the test environment is co-located in the same data center as the disaster recovery hardware, repurposing the hardware is smooth and takes less time. For an unplanned disaster scenario, the MSIT SAP team follows a documented disaster recovery process and communication procedure.

# Minimizing Planned Downtime

To minimize overall application downtime, it is important to adequately plan for maintaining uptime during certain maintenance activities, such as software upgrades, security patching, server upgrades and exchanges, storage swaps, and deployment of SAP support packages. It is important to note that the techniques used to reduce planned downtime are *very* similar to the processes followed for unplanned downtime. For planned downtime, MSIT defines estimates of the duration of specific activities in order to set appropriate outage expectations to business and internal end users. Each of the following processes and activities described in the next section details the estimated timelines that MSIT uses along with the steps used to achieve high availability.

## Security Upgrading or Updating (Operating System and Components)

This section describes how MSIT patches the operating system and other components that are not related to SQL Server, as well as applications on the server that hosts the SAP ERP database.

Table 1 details the patching process along with the associated timeline of hours before and after the planned downtime.

|  |  |
| --- | --- |
| Timeline | Description |
| Several days before downtime | Apply patches on the log shipping secondary.  Apply patches on the witness. |
| 4 hours before downtime | Apply security patches to both the principal and mirror database server during uptime. These updates relate to the operating system or security. Because this process applies to components that are not related to SQL Server, the sequence of operation against the mirror and principal database servers is less of a consideration. Applying changes to the mirror server first does have the advantage of avoiding downtime in the event of unexpected failures. After the patches are installed, the servers are *not* rebooted. Rebooting takes place instead during the last step of this process. |
| 2 hours before downtime | Reboot the mirror server (the principal database server is still online – no experienced downtime). |
| Planned failover  (brief planned downtime window) | After rebooting the mirror server, initiate a database mirroring failover on the principal, thus switching database mirroring roles. This step is performed within a planned downtime window, even though the actual downtime is very brief – usually a few seconds to a minute. Initiate a failover during downtime to avoid affecting any active transactions or batch jobs.  The SAP application automatically reconnects work processes to SQL Server if the connection is closed or dropped due to the failover, and it is essentially redirected to the new principal after a database failover is triggered. |
| Immediately after downtime | Reboot the new mirror server. Because the principal database is not affected, no downtime is incurred. |

**Table 1:** Security Patching Process (operating system and components)

## SQL Server Upgrades or Patching

This section details the process that MSIT follows for applying SQL Server upgrades, service packs, and cumulative updates. In order to further minimize overall downtime, this process is combined with a quarterly release change schedule for SAP Support Packages. Table 2 details the process for upgrading or applying a service pack or cumulative update package across production and disaster recovery SQL Server instances used to host the SAP ERP database.

|  |  |
| --- | --- |
| Timeline | Description |
| One day before scheduled downtime | Upgrade the log shipping secondary at the disaster recovery site. This is performed one day before the planned downtime. You can perform an upgrade on the log-shipped instance of SQL Server because SQL Server transaction logs from a lower version can be restored on an up-level instance of SQL Server.  *Warning: If a failover is necessary to the disaster recovery site after this step, the database is upgraded upon recovery to the up-level version (if upgrading or applying a service pack/cumulative update). After this happens and production activity is routed to the disaster site, you cannot revert to a down-level version of SQL Server.*  In this step, you also upgrade the witness. |
| 4 hours before downtime | Upgrade the SQL Server instance hosting the mirror database in the primary data center. After you upgrade the SQL Server instance, the database mirroring session will not be impacted and will continue to run.  *Warning: If a database mirroring failover occurs after this step, the database is immediately upgraded to the higher version level of SQL Server, eliminating the ability to return to a down-level version on mirror partner.* |
| 1 or more hours before downtime | Upgrade the SQL Server Native Client (SNAC) on the application server for the SAP Central Instance.This step does not require a reboot and is very fast. The setup/upgrade tool does not detect the SNAC component for upgrades, so you must manually uninstall and then reinstall it. |
| Planned downtime | Fail over the SAP ERP database to the upgraded mirror. MSIT estimates between 10 and 15 minutes of downtime in this step, which is the time it takes to execute SQL Server upgrade scripts. If you are performing a version upgrade, change the database compatibility mode for the SAP ERP database. |
| 15 minutes after downtime | Upgrade the SQL Server instance that hosts the SAP ERP mirror (the old principal).  *Note that this step should happen as soon as possible after the principal database is upgraded. Until the mirror database is upgraded, no transaction log backup can be applied on the mirror, and mirroring will be in a suspended state. As soon as the mirror database is upgraded, the mirroring session will resume.* |

**Table 2:** SQL Server Upgrade and Patching Process

## Storage Upgrades

This section details the process that MSIT follows to perform SAN configuration, storage changes, or component swaps that require outage periods against the storage used by the SAP ERP database. Because the principal and mirror are on separate independent SAN storage, the storage on these servers can be upgraded independently, and the application downtime is limited to database mirroring failover.

|  |  |
| --- | --- |
| Timeline | Description |
| 5 days before downtime | In preparation for the SAN configuration change, remove the SAP ERP database mirroring session. This is done because if changes are required to the disk, a database cannot be reattached using WITH NORECOVERY on the mirror.  Hand off the SQL Server instance hosting the mirror database to the data center personnel for maintenance work during which time SAN modifications can be made. During this time the production database server remains online. You can add a third server to the topology that is used as a temporary mirror. Doing this helps maintain high availability during the storage maintenance procedure. |
| 4 days and 20 hours prior to downtime | After completion of SAN changes, the server should be returned for use. At this point, you should perform a full backup on the principal, copy to the mirror server, and then restore using WITH NORECOVERY in order to reestablish a mirroring session again. If a third server was used temporarily for database mirroring, the mirroring session to the temporary server must be removed before the original mirror server is reestablished. |
| Planned downtime | Fail over the SAP ERP database (role switch) to the mirror SQL Server instance. This is only a very brief downtime (typically a few seconds) and SAP systems remain available during that time. The principal database should now be running on the upgraded SAN storage. |
| 2 days after downtime | In preparation for the SAN configuration change on the remaining database server, remove the database mirroring session.  Hand off the SQL Server instance hosting the mirror database to the data center personnel for maintenance work during which time SAN modifications are made. During this time the production database server incurs no downtime.  You can also increase high availability during the storage maintenance procedure by adding a third server to the topology. This server can be used as a temporary mirror. |
| 2 days and 4 hours after downtime | After completion of SAN changes, the server should be returned for use. At this point, you should perform a full backup on the principal, copy it to the mirror server, and then restore using WITH NORECOVERY in order to reestablish a mirroring session again.  If a third server was temporarily used for database mirroring, the mirroring session must be removed prior to restoring the original instance of SQL Server back into the topology. |

**Table 3:** Storage Upgrade Process

For some types of storage upgrades or changes, you may not need to remove mirroring:

* If the downtime is short enough that there is sufficient log space for accumulated log entries. If the only concern is log space on the principal, database mirroring can be converted to log shipping temporarily during the storage changes.
* If the SAN change or maintenance leaves data files intact in their original locations.

In such cases, you can stop SQL Server for the duration of SAN maintenance, and starts it up again upon completion.

MSIT also has the potential option of either a live migration or a copy to new SAN storage, which they then use for some of Microsoft’s smaller databases. These are databases where and storage changes where the files and folder structures remain unchanged.

## Server Swaps

The following table details the MSIT process for exchanging an existing SAP ERP database server with a new database server. Table 4 illustrates the steps for swapping the mirror server.

|  |  |
| --- | --- |
| Timeline | Description |
| 1 day before the new server becomes active | Provision a *new* database server and attach it to the SAN storage. Install SQL Server and apply the appropriate service packs, bringing the new server to the same level of service packs as the existing production SQL Server instance.  Perform a full backup of the primary production SAP ERP database. |
| 4 hours prior to new server becoming “active” | Copy the full backup to the new database server and then restore (without recovery). |
| New server deemed active | Configure log shipping and establish between the primary SAP ERP database and the new database server. After log shipping synchronization, temporarily stop transaction log backups. |
| 1 hour and 30 minutes after new server is active | Remove the existing database mirroring session. After that, you can remove the mirror server from the topology. |
| 1 hour and 45 minutes after new server is active | Configure a new database mirroring session between the primary database server and the new database server. |
| 2 hours after the new server is “active” | Resume transaction log backups on the primary (principal) database server. The new database server now actively functions as the mirror server partner. |

**Table 4:** Server Swap Process, Part I

The aforementioned process illustrates the replacement of a single database server (in this case – replacement of the mirror server). If *both* servers need replacement (both mirror and principal), the previous process is extended with the steps detailed in Table 5.

|  |  |
| --- | --- |
| Timeline | Description |
| Planned downtime | Fail over the principal database to the new database server. This downtime is typically a few seconds, and doesn’t require an SAP outage. |
| 5 minutes after downtime | Exchange the former primary database server, which is now the mirror server, with a new server as described in Table 4. |

**Table 5:** Server Swap Process, Part II

## SAP Support Packages

The following table details MSIT’s process for rolling out major SAP changes, such as those made via SAP Support Packages. This process creates two independent versions of production database that are readily available on disk, enabling the SAP ERP team to roll back changes if necessary, all with minimal downtime in case MSIT decides not to implement the changes immediately.

|  |  |
| --- | --- |
| Timeline | Description |
| Application downtime begins | Suspendthe SQL Server database mirroring session for the SAP ERP database. |
| Five minutes after application downtime begins | Major changes to the SAP system, such as SAP Support Packages, can then be applied to the primary SAP ERP database (the principal database). |

**Table 6:** SAP Support Package Process, Part I

After applying the major change, MSIT decides whether the upgrade was successful. If the upgrade is deemed successful, the database mirroring session is resumed. If it is not deemed successful, the following steps are performed.

|  |  |
| --- | --- |
| Timeline | Description |
| 15 minutes after decision | If changes must be reverted, remove the database mirroring session and stop activity against the primary database (the SAP ERP database involved in the major change). |
| 30 minutes after decision | The formerly mirror and unmodified database should then be brought online (restore with recovery) and will become the *new* SAP ERP primary database. The SAP application should then be pointed to this database and the application work resumed. |
| 45 minutes after decision | The major upgrade change should then be delayed until all major problems are fixed. In the meantime, a new database mirroring session must be established between the new principal and the formerly primary database server. |

**Table 7:** SAP Support Package Process, Part II

# Monitoring

MSIT monitors all SAP ERP database servers across production, disaster recovery, and test environments. The following technologies are used for monitoring, alerting, and responding to issues:

* Microsoft Operations Manager 2005 is used for SAP ERP to alert on errors in the Windows Event logs and identify server availability issues. ***Note****: System Center Operations Manager 2007 is used by other MSIT teams at this time, and it will eventually be used for SAP ERP.*
* SQL Server Agent jobs are used for the following purposes:
  + MSIT has created a custom database mirroring SQL Server Agent job and alert. This job and alert raises Windows events and SQL Server error log events if the database mirror is out of sync past a user-defined threshold. An alert is also raised if the mirror is identified to be in a disconnected state. This job runs every 3 minutes and looks for an out-of-sync threshold of 10 minutes (this number is sometimes increased if any special processing is underway such as data compression or index rebuilds, or if any known network issues are occurring). The script checks the last log sequence number (LSN) applied to the mirror and then determines the approximate out-of-sync duration against what transaction log backup the LSN is contained in. This custom SQL Server Agent job is included later in this paper, in Appendix B.
  + Another SQL Server Agent job is used to enable and disable SQL Server Agent jobs based on the current role of the database (principal or mirror). The specified job runs every minute and searches for a change in the SAP ERP database state (for example – if the database switches to an unrecovered state). For more information about the script and methods used to achieve the job toggling, see the blog[Dealing with Scheduled Tasks in a SQL Server Environment using Database Mirroring](http://blogs.msdn.com/saponsqlserver/archive/2008/08/23/dealing-with-scheduled-tasks-in-a-sql-server-environment-using-database-mirroring.aspx) (<http://blogs.msdn.com/saponsqlserver/archive/2008/08/23/dealing-with-scheduled-tasks-in-a-sql-server-environment-using-database-mirroring.aspx>)*.*
  + The MSIT team also uses a set of log shipping SQL Server Agent jobs on both the principal and mirror SQL Server instances. Copies of transaction logs are stored redundantly on both the principal and mirror servers. In addition to this they have a cleanup job for redundant backup files (transaction log backup files on the mirror that were not deleted automatically) on the mirror, because the log shipping restore job normally manages file retention and cleanup.
  + The MSIT SAP ERP team also has the following alerting mechanisms:
    - A log shipping SQL Server Agent job that manages out-of-sync alerts. This job checks to see when the last transaction log restore occurred. It runs on the server that contains the log shipped replica of the database.
    - A database mirroring standard alert job that executes the system stored procedure [sp\_dbmmonitorupdate](http://msdn.microsoft.com/en-us/library/ms403827.aspx).
    - A SQL Server Agent job that monitors for sustained blocking events.

MSIT also monitors for, reports on, and sends alerts when it encounters specific SAP ERP application states such as SAP update transaction failures, short dump error counts, and application queue problems. Because users experience operational problems as availability issues, the MSIT SAP ERP team is also working to make more information about functional health available by expanding monitoring and reporting. To do this, they plan to use alerts based on test SAP transactions and SAP logon tests. This monitoring will likely be conducted outside the scope of SQL Server and will be managed via the middle-tier SAP application tier.

# Conclusion

Since deploying SAP ERP on SQL Server 2005 and then upgrading to SQL Server 2008, the MSIT SAP ERP team has achieved high levels of availability and provided a robust solution for disaster recovery using native SQL Server features such as database mirroring and log shipping. MSIT also uses separate local SANs in the primary data center, as well as providing a second geographically dispersed data center for use with disaster recovery. Microsoft’s SAP ERP data tier production, disaster recovery, and test environments all have identical hardware and SAN disk configurations. The architectural solution, aided by good operational practices, such as continuous monitoring, documented procedures, and an annual disaster recovery exercise, has resulted in minimal planned and unplanned downtime.

# Appendix A: Server and Storage Specifications

## Servers

The MSIT SAP ERP team has standardized on HP DL580 G5 for all servers running SQL Server. All servers (the principal, the mirror, and the log shipping secondary) are configured identically. Each server includes redundant components such as multiple NICs and multiple HBAs in order to minimize a single point of failure within the server enclosure. Each server has:

* 4 sockets, each socket with six cores @2.9 GHz
* 96 GB or RAM
* x64 architecture

## Storage

An EMC CX3-80 SAN is used for each database server. One SAN is connected to the principal database server and the second SAN is connected to the mirror partner database server. This eliminates a single point-of-failure in the event of a single SAN outage. Each storage array is dedicated solely to the corresponding SQL Server instance, and it is not shared with any other applications, preventing performance overhead and minimizing factors that could drive unpredictable I/O performance.

For redundancy within the storage arrays, the data drives and the transaction log drives are configured with RAID 1+0. The database servers are connected to the SAN storage via two HBAs. Sixteen mount points are mapped to the data drive, on which the SAP ERP database is then broken into 32 data files.

As a best practice, storage on all database servers is configured identically.

# Appendix B: Scripts

## The sp\_MirroringStatus Procedure

MSIT uses the following stored procedure to validate the status of a database mirroring session. This procedure is executed every three minutes in production. The first input parameter is the database name, and the second input parameter is the delay threshold, in minutes. The procedure calculates the LSN delay based on the log backup set that contains the LSN and raises windows and error log events.

\*\* Please note that this is a sample script and assumes only one user database. If you want to extend this procedure to support multiple databases, you will need to modify the predicate accordingly. \*\*

--' THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF

--' ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO

--' THE IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A

--' PARTICULAR PURPOSE.

--'

--' IN NO EVENT SHALL MICROSOFT AND/OR ITS RESPECTIVE SUPPLIERS

--' BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY

--' DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS,

--' WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS

--' ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE

--' OF THIS CODE OR INFORMATION.

create procedure [dbo].[sp\_MirroringStatus] @Database nvarchar(256),

@Delay\_threshold int = 10

as

--

-- Init and Globals

--

declare @mirrorLsn numeric(25,0)

declare @warningMsg varchar(max)

declare @errorMsg varchar(max)

declare @min\_seconds\_behind varchar(25)

declare @min\_minutes\_behind numeric(25,1)

declare @max\_seconds\_behind varchar(25)

declare @max\_minutes\_behind numeric(25,1)

declare @mirroring\_state varchar(60)

declare @backup\_set\_id int

declare @first\_lsn numeric(25,0)

declare @last\_lsn numeric(25,0)

declare @backup\_start\_date datetime

declare @backup\_finish\_date datetime

declare @backup\_size nvarchar(256)

declare @physical\_device\_name nvarchar(510)

set @mirrorLsn = 0

set @warningMsg=''

set @errorMsg=''

--

-- Get the Mirroring Lsn

--

select @mirrorLsn = mirroring\_failover\_lsn from sys.database\_mirroring where database\_id=db\_id(@Database)

--

--

if @mirrorLsn > 0

begin

select top 1 @last\_lsn=last\_lsn, @backup\_start\_date=backup\_start\_date from msdb..backupset where type='L' order by backup\_set\_id desc

if @mirrorLsn > @last\_lsn

begin

set @min\_minutes\_behind=0

select @max\_seconds\_behind = datediff(second,@backup\_start\_date,getdate())

select @max\_minutes\_behind = convert(numeric(25,1),@max\_seconds\_behind) / 60

print ''

print 'MIRRORING LSN OK: Mirror LSN is newer than the last TLOG backup LSN.'

if @max\_minutes\_behind=0

begin

print 'MIRRORING DELAY TIME: Between 0 and ' + @max\_seconds\_behind + ' seconds.'

end

else

print 'MIRRORING DELAY TIME: Between 0 and ' + convert(varchar(27),@max\_minutes\_behind) + ' minutes.'

--Check Delay Threshold specified against actual Delay Time

if @max\_minutes\_behind >= @Delay\_threshold

begin

if exists (select spid from master..sysprocesses where cmd = 'BACKUP LOG')

set @errorMsg='MIRRORING DELAY ERROR: ' + convert(varchar(27),@Delay\_threshold) + ' MINUTE DELAY THRESHOLD EXCEEDED FOR DATABASE ' + @Database + '!' + char(10) +

'LONG TLOG BACKUP JOB: ' + @Database + ' TRANSACTION LOG BACKUP STILL RUNNING.' + char(10)

else

set @errorMsg='MIRRORING DELAY ERROR: ' + convert(varchar(27),@Delay\_threshold) + ' MINUTE DELAY THRESHOLD EXCEEDED FOR DATABASE ' + @Database + '!' + char(10)

end

else

begin

print 'MIRRORING DELAY OK: ' + convert(varchar(27),@Delay\_threshold) + ' minute delay threshold greater than Delay Time for database ' + @Database + '.'

end

end

else

begin

declare backupset\_cursor cursor

for select s.backup\_set\_id,

s.first\_lsn,

s.last\_lsn,

s.backup\_start\_date,

s.backup\_finish\_date,

s.backup\_size,

m.physical\_device\_name

from msdb..backupset s,

msdb..backupmediafamily m

where type='L' and

s.media\_set\_id=m.media\_set\_id

order by s.backup\_set\_id desc

open backupset\_cursor

fetch next from backupset\_cursor

into @backup\_set\_id,

@first\_lsn,

@last\_lsn,

@backup\_start\_date,

@backup\_finish\_date,

@backup\_size,

@physical\_device\_name

while @@fetch\_status=0

begin

if @mirrorLsn >= @first\_lsn and @mirrorLsn <= @last\_lsn

begin

print ''

print 'Mirror LSN: ' + convert(varchar(25),@mirrorLsn)

print 'TLOG File: ' + @physical\_device\_name

/\*

select @backup\_set\_id as 'backup\_set\_id',

@first\_lsn as 'first\_lsn',

@last\_lsn as 'last\_lsn',

@backup\_start\_date as 'backup\_start\_date',

@backup\_finish\_date as 'backup\_finish\_date',

@backup\_size as 'backup\_size'

\*/

print ''

select @min\_seconds\_behind = datediff(second,@backup\_start\_date,getdate())

select @min\_minutes\_behind = convert(numeric(25,1),@min\_seconds\_behind) / 60

--Get start time of previous log backup to determine max delay time

fetch next from backupset\_cursor into @backup\_set\_id, @first\_lsn, @last\_lsn, @backup\_start\_date, @backup\_finish\_date, @backup\_size, @physical\_device\_name

select @max\_seconds\_behind = datediff(second,@backup\_start\_date,getdate())

select @max\_minutes\_behind = convert(numeric(25,1),@max\_seconds\_behind) / 60

print 'MIRRORING DELAY TIME: Between ' +

convert(varchar(27),@min\_minutes\_behind) + ' and ' +

convert(varchar(27),@max\_minutes\_behind) + ' minutes.'-- + char(10)

break

end

fetch next from backupset\_cursor into @backup\_set\_id, @first\_lsn, @last\_lsn, @backup\_start\_date, @backup\_finish\_date, @backup\_size, @physical\_device\_name

end

close backupset\_cursor

deallocate backupset\_cursor

end

select @mirroring\_state=mirroring\_state\_desc from sys.database\_mirroring where database\_id=db\_id(@Database)

--Check sys.databases mirroring\_state\_desc. - Must be 'SYNCHRONIZED'.

if @mirroring\_state<>'SYNCHRONIZED'

begin

if @mirroring\_state='SUSPENDED'

begin

set @warningMsg='MIRRORING STATUS WARNING: ' + @Database + ' database Mirroring ' + @mirroring\_state + ' between ' +

convert(varchar(27),@min\_minutes\_behind) + ' and ' +

convert(varchar(27),@max\_minutes\_behind) + ' minutes ago.'

end

else

begin

set @errorMsg=@errorMsg + 'MIRRORING STATUS ERROR: ' + @Database + ' database Mirroring ' + @mirroring\_state + '! '

end

end

else

begin

if @min\_minutes\_behind > @Delay\_threshold

begin

set @errorMsg='MIRRORING DELAY ERROR: ' + convert(varchar(27),@Delay\_threshold) + ' MINUTE DELAY THRESHOLD EXCEEDED FOR DATABASE ' + @Database + '! '

--set @errorMsg=@errorMsg + 'MIRROR STATUS INCORRECT: Status is ''SYNCHRONIZING'' but Mirroring is behind. Monitor progress..'

end

else print 'MIRRORING STATUS OK: ' + @Database + ' database Mirroring ' + @mirroring\_state + '.'

end

end

else

begin

set @warningMsg = N'WARNING: Mirroring not active for database ' + @Database

end

-- Raise Warning if warning message found

if @warningMsg<>''

begin

print ''

raiserror (@warningMsg, 10, 1) with log

end

-- Raise error if status error found or delay threshold exceeded.

if @errorMsg<>''

begin

set @errorMsg = @errorMsg + 'MIRRORING DELAY TIME: Between ' +

(case(convert(varchar(27),@min\_minutes\_behind)) when '0.0' then '0' else convert(varchar(27),@min\_minutes\_behind) end) + ' and ' +

convert(varchar(27),@max\_minutes\_behind) + ' minutes.'

print ''

raiserror (@errorMsg, 19, 1) with log

end

**For more information:**

<http://www.microsoft.com/sqlserver/>: SQL Server Web site

<http://technet.microsoft.com/en-us/sqlserver/>: SQL Server TechCenter

<http://msdn.microsoft.com/en-us/sqlserver/>: SQL Server DevCenter

Did this paper help you? Please give us your feedback. Tell us on a scale of 1 (poor) to 5 (excellent), how would you rate this paper and why have you given it this rating? For example:

* Are you rating it high due to having good examples, excellent screen shots, clear writing, or another reason?
* Are you rating it low due to poor examples, fuzzy screen shots, or unclear writing?

This feedback will help us improve the quality of white papers we release.

[Send feedback](mailto:sqlfback@microsoft.com?subject=White%20Paper%20Feedback:%20[High%20Availability%20and%20Disaster%20Recovery%20for%20Microsoft’s%20SAP%20Data%20Tier]).