# 

**AlwaysOn Architecture Guide: Building a High Availability and Disaster Recovery Solution by Using AlwaysOn Availability Groups**

SQL Server Technical Article

**Authors:** Joseph Sack (SQLskills.com), Sanjay Mishra (Microsoft)

**Technical Reviewers:** Lindsey Allen (MS), Juergen Thomas (MS), Mike Weiner (MS), Prem Mehra (MS), Yorihito Tada (MS), Curt Matthews (MS), Amitabh Tamhane (MS), Aditya Samant (MS), Daniel Janik (MS), Jimmy May (MS), David P Smith (ServiceU), Richard Waymire (SolidQ), Brent Ozar (Brent Ozar PLF), Wolfgang Kutschera (bwin.party), Paul S. Randal (SQLskills.com), Gianluca Hotz (SolidQ), Ayad Shammout (Caregroup)

**Content Program Manager:** Glenn Minch (Microsoft)

**Published:** June 2012

**Applies to:** SQL Server 2012

**Summary:** SQL Server 2012 AlwaysOn Availability Groups provides a unified high availability and disaster recovery (HADR) solution that improves upon legacy functionality previously found across disparate features. Prior to SQL Server 2012, several customers used database mirroring to provide local high availability within a data center, and log shipping for disaster recovery across a remote data center. With SQL Server 2012, this common design pattern can be replaced with an architecture that uses availability groups for both high availability and disaster recovery. This paper details the key topology requirements of this specific design pattern, including quorum configuration considerations, steps required to build the environment, and a workflow that shows how to handle a disaster recovery event in the new topology.

Copyright

This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2012 Microsoft. All rights reserved.

Contents

[Introduction 4](#_Toc322535190)

[Legacy Architecture: Database Mirroring for High Availability and Log Shipping for Disaster Recovery 4](#_Toc322535191)

[AlwaysOn Availability Groups for High Availability and Disaster Recovery 5](#_Toc322535192)

[Deployment Planning and Considerations 6](#_Toc322535193)

[Topology Prerequisites 6](#_Toc322535194)

[Quorum Model 7](#_Toc322535195)

[Configuring the WSFC Quorum mode 9](#_Toc322535196)

[Using DMVs and AlwaysOn Dashboard to view Quorum Information 9](#_Toc322535197)

[Configuring Quorum Node Votes 11](#_Toc322535198)

[Client Connectivity 12](#_Toc322535199)

[Legacy Database Mirror Connection Strings 12](#_Toc322535200)

[Availability Group Listener 12](#_Toc322535201)

[Multi-Subnet Connection Support 13](#_Toc322535202)

[Building the Availability Group Solution 13](#_Toc322535203)

[Configuring Availability Groups for local High Availability and remote Disaster Recovery 13](#_Toc322535204)

[Monitoring Considerations 21](#_Toc322535205)

[Recovering from a Disaster 22](#_Toc322535206)

[Reverting Back to the Primary Data Center 25](#_Toc322535207)

[Conclusion 29](#_Toc322535208)

[References 29](#_Toc322535209)

# Introduction

Microsoft SQL Server 2012 AlwaysOn provides flexible design choices for selecting an appropriate high availability (HA) and disaster recovery (DR) solution for your application. There are multiple [design patterns for building SQL Server 2012 AlwaysOn HA and DR solutions](http://sqlcat.com/sqlcat/b/msdnmirror/archive/2011/12/22/sql-server-2012-alwayson-high-availability-and-disaster-recovery-design-patterns.aspx). This white paper describes a solution that uses AlwaysOn Availability Groups for HA and DR. This is a solution that is based purely on non-shared storage, because each instance of SQL Server in the topology has its own copy of data, and does not need to share storage. For more information about other design choices, see [SQL Server 2012 AlwaysOn High Availability and Disaster Recovery Design Patterns](http://go.microsoft.com/fwlink/?LinkId=255048).

Prior to SQL Server 2012, a common HA and DR deployment architecture involved the use of database mirroring for local high availability and log shipping for remote disaster recovery. With SQL Server 2012, an availability groups solution with multiple secondaries can replace the legacy solution that uses database mirroring and log shipping.

This paper covers planning considerations and walks through the steps required to build the availability groups for high availability and disaster recovery requirements. This paper also details the steps required to recover from a disaster, and it discusses how to revert to the primary data center once the primary data center is restored.

This paper assumes a basic knowledge of AlwaysOn Availability Groups, high availability, and disaster recovery concepts. For more information about the full AlwaysOn solution feature set, see the [Microsoft SQL Server AlwaysOn Solutions Guide for High Availability and Disaster Recovery](http://msdn.microsoft.com/library/hh781257.aspx) white paper. The target audience for this white paper includes operational SQL Server database administrators and technology architects. This paper is also appropriate for system administrators who collaborate with database administrators to manage Windows Server, Active Directory Domain Services (AD DS), Windows Server Failover Clusters (WSFC), and networking.

# Legacy Architecture: Database Mirroring for High Availability and Log Shipping for Disaster Recovery

Prior to SQL Server 2012, one popular customer SQL Server deployment architecture involved using database mirroring for high availability within the primary data center, and using log shipping for cross-data center disaster recovery. For this solution, database mirroring is configured within the primary data center. To achieve automatic failover, synchronous database mirroring with a witness (a third SQL Server instance) is configured. When zero data loss is required, the database mirroring high-safety mode (synchronous) setting is enabled to help ensure zero data loss between the two servers located in the primary data center. To improve database availability within the primary data center, a third SQL Server instance is configured to act as a witness to enable automatic failover between the database mirroring partners.

If a primary data center outage renders both database mirroring partner instances unavailable, log shipping is used for disaster recovery. Log shipping involves ongoing transaction log backups of the principal database. These transaction log backups are then copied to a SQL Server instance in the disaster recovery data center. Incoming transaction log backups are restored in sequence on an ongoing basis. You could also choose to configure log shipping for read-only workloads, but with the drawback that read-only connections must be disconnected before incoming transaction log backups are applied. Figure 1 shows a representation of this solution architecture.

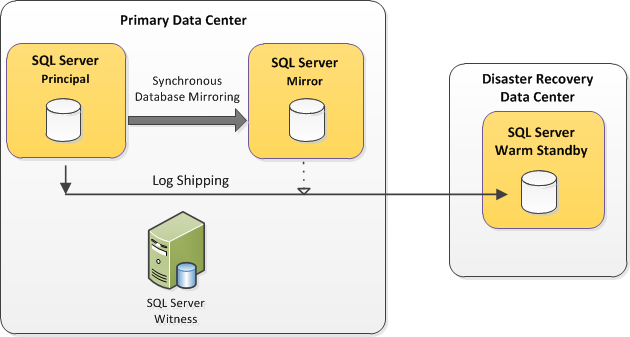


Figure : Database mirroring for high availability and log shipping for disaster recovery

For more information about this specific solution, including a practical example, see the technical case study [High Availability and Disaster Recovery for Microsoft’s SAP Data Tier: A SQL Server 2008 Technical Case Study](http://go.microsoft.com/fwlink/?LinkId=255179).

# AlwaysOn Availability Groups for High Availability and Disaster Recovery

AlwaysOn Availability Groups can be used to replace the previously described database mirroring and log shipping solution. Using availability groups for HA and DR provides the following advantages:

* You can group more than one user database into a single unit of failover. In contrast, database mirroring permits only one user database as a unit of failover.
* Availability groups multiple secondaries enable the user to unify the HA and DR solution into one technology, instead of using multiple technologies used in the earlier solution.
* Secondary replicas can also be configured to permit read-only workloads to obtain close to real-time data from them. Unlike with log shipping, ongoing read-only connections to the secondary replicas do not need to be disconnected in order to see ongoing data modifications against the primary replica. Secondary replicas can also be used to offload full database and transaction log backup operations.
* Availability groups and the associated availability group listener support automatic client redirection to the primary replica or redirection to available readable secondaries. Availability group listeners remove the need to designate a failover partner in the client connection string.

Figure 2 shows the HA and DR solution using availability groups.

Figure : Using availability groups for high availability and disaster recovery

As Figure 2 indicates, the three nodes, each running an instance of SQL Server, participate in a *single* Windows Server Failover Cluster (WSFC) that spans two data centers. Also, it is important to note that this is a non-shared solution, and the nodes don’t share any storage with another node. Each node is running an instance of SQL Server and has its own copy of the data.

**Note**: Figure 2 illustrates a simple scenario with two data centers: the primary data center hosts two replicas, and the DR data center hosts one replica. The architecture allows for variations to this topology using multiple data centers as well as multiple replicas (up to five). The discussion in this white paper focuses on the topology shown in Figure 2; however, the general concepts apply to the other variations as well. One such variation, which involves three data centers, is discussed in Appendix A.

# Deployment Planning and Considerations

The next few sections detail key planning considerations, requirements, and prerequisites you should consider when planning the deployment of availability groups for high availability and disaster recovery.

## Topology Prerequisites

It is important to understand prerequisites and restrictions before you design an HA+DR solution using availability groups. For more information about prerequisites for and restrictions on using availability groups, see [Prerequisites, Restrictions, and Recommendations for AlwaysOn Availability Groups (SQL Server)](http://msdn.microsoft.com/library/ff878487(v=sql.110).aspx).

## Unit of Failover

In this HA+DR solution, the unit of failover is the availability group (a group of user databases). SQL Server Agent jobs, logins, linked servers, and other objects that are stored outside of the availability databases do *not* fail over with the availability group. Consider the use of [contained databases](http://technet.microsoft.com/en-us/library/ff929071.aspx) for containing logins that fail over across the availability replicas. For other objects outside of the user database such as SQL Server Agent jobs, linked servers, and SQL Server Integration Services packages, you will need to take additional synchronization steps across the SQL Server instances.

## Considerations for Replacing Log Shipping

If you replace an earlier solution involving log shipping with an availability groups solution, keep the following considerations in mind:

* Removing log shipping means that there is no “delayed apply” functionality on the secondary replicas. Availability group replica log records are applied immediately and the corresponding delayed apply that log shipping offered is not available for availability groups. If this functionality was a required part of your legacy solution, you’ll need to plan for an alternative solution or use log shipping in conjunction with your availability group solution.
* Removing log shipping also means that the regular log backup job is removed. *Availability groups are not a replacement for a backup and restore strategy*. You will need to re-establish periodic log backups as a separate process for availability groups in order to keep up the essential management of the transaction log for each availability database in the availability group.

## Quorum Model and Node Votes

**Note**: The quorum and related discussions in this white paper apply to the solution running on Windows Server 2008 and Windows Server 2008 R2 operating systems, with appropriate software updates.

Because the underlying infrastructure of an availability group is a WSFC, it is important to consider appropriate quorum model for the WSFC. Quorum configuration is managed at the WSFC level, irrespective of the number of replicas and the number of availability groups hosted in the WSFC.

WSFC supports four quorum models. However, not all quorum models are appropriate for a *non-shared-storage* solution as is the scope of this white paper. As such, the shared disk based quorum models (Node and Disk Majority, No Majority: Disk Only) are not applicable. This leaves us with two quorum models for our scoped solution architecture: Node and File Share Majority or Node Majority. For more information about the four quorum models, see [Failover Cluster Step-by-Step Guide: Configuring the Quorum in a Failover Cluster](http://technet.microsoft.com/library/cc770620(v=WS.10).aspx).

It is important to take into consideration the number of voting nodes before you select a quorum model. Assigning appropriate node votes plays an important role in the HA+DR design. By default, every node in a WSFC has a vote, but that may not be appropriate for your particular HA+DR solution, depending upon the distribution nodes in the primary and DR data center. There is a Windows Server hotfix available (<http://support.microsoft.com/kb/2494036>) that allows you to assign 1 vote to some nodes and 0 votes to some other nodes in the WSFC. The NodeWeight property of the WSFC node represents the vote for that particular node. The value ‘0’ means the node doesn’t have a vote. The value ‘1’ means the node has a quorum vote. This hotfix must be installed on each node in the topology.

General recommendations for quorum voting for an AlwaysOn HA+DR solution are provided in the [Recommended Adjustments to Quorum Voting](http://msdn.microsoft.com/library/hh270280#RecommendedAdjustmentstoQuorumVoting) section of [WSFC Quorum Modes and Voting Configuration](http://msdn.microsoft.com/library/hh270280) in SQL Server Books Online. These should be treated as guidelines for deciding on the voting scheme for the AlwaysOn solution. Taking these guidelines into consideration, for the availability groups HA+DR solution presented in Figure 2, the voting scheme will be:

* One vote to each node in the primary data center
* Zero votes to the node in the disaster recovery data center

This vote assignment helps ensure that the quorum of the nodes in the primary data center is not compromised by outages in the DR data center or loss of connectivity between the two data centers.

If you have an odd number of voting nodes, using the node majority quorum model is the best choice. Because the topology discussed in this white paper has even number of voting nodes (the two nodes in the primary data center), you can choose either of the two following options:

* Add an additional voting node to the WSFC in the primary data center, and then use the Node Majority quorum model. This additional node does not does not need to have SQL Server installed on it or be a replica of the availability group. This configuration with appropriate node votes is shown in Figure 3 (how to assign node votes will be discussed later in the white paper.



Figure : Node vote assignment for an HA+DR AG availability group deployment with the Node Majority quorum model

* Use the Node and File Share Majority quorum model with a protected file share witness. The file share provides an additional vote to establish quorum, and it does not contain any SQL Server data. This configuration with appropriate node votes is shown in Figure 4.

Figure : Node vote assignment for an HA+DR availability group deployment with the Node and File Share Majority quorum model

Note that the file share witness is outside of the WSFC hosting the availability group. A given file share can act as a witness to one or more WSFCs. The file share witness, if used, always has a vote. You can’t assign 0 votes to a file share witness.

The quorum model and vote assignments presented in Figure 3 and Figure 4 assume that you have three replicas (one primary and two secondaries) of the availability group (two replicas in the primary data center, and one replica in the DR data center). If you have a different number of nodes and replicas, the vote assignments may be slightly different, but the basic principles still apply. For example, if you have an additional replica in the primary data center (to offload read or backup workload), you will have a total of three nodes in the primary data center, and hence can use Node Majority quorum model and don’t need the additional node (as illustrated in Figure 3) or the file share witness (as illustrated in Figure 4). And in that scenario, you assign one vote to each node the primary data center, and zero votes to the node in the DR data center.

The quorum model and vote assignments presented in Figure 3 and Figure 4 also assume that the solution spans two data centers. If you have more data centers, and you plan to put some part of your solution in a third data center, the quorum model decisions and vote assignments may vary.

### Tools to View or Change Quorum Model and Node Votes

There are multiple ways to view and change the cluster quorum model and the quorum votes. The following tables list the various tools for these tasks.

|  |  |
| --- | --- |
| To view quorum model | To change quorum model |
| Windows Failover Cluster Manager  Windows PowerShell Cluster.exe SQL Server dynamic management views (DMVs) AlwaysOn Dashboard in SQL Server Management Studio | Windows Failover Cluster Manager  Windows PowerShell Cluster.exe |

|  |  |
| --- | --- |
| To view node votes | To change node votes |
| Windows PowerShell Cluster.exe SQL Server dynamic management views (DMVs) AlwaysOn Dashboard | Windows PowerShell Cluster.exe |

### Configuring the WSFC Quorum Model

Below are examples of using Windows PowerShell via command line to view the current quorum model, and to change the quorum model.

**To view the existing quorum model**

Get-ClusterQuorum

**To configure Node Majority Quorum Model**

Set-ClusterQuorum -NodeMajority

**To change the quorum model to Node and File Share Majority**

Set-ClusterQuorum -NodeAndFileShareMajority \\EMU-DC\Witness

The witness file share you choose must *not* be on a node that already participates in the AlwaysOn WSFC configuration. However, it can be placed as a share on another WSFC configuration. It must exist within the same Active Directory domain as the WSFC. Also, the WSFC cluster service account requires read and write permissions to the file share witness. Failover Cluster Manager has built-in logic to add these permissions to the file share witness as long as the administrative account that is changing the quorum model has permissions on the file share.

### Using DMVs and AlwaysOn Dashboard to View Quorum Information

Though you cannot set or change the quorum model or node votes through SQL Server tools, you can use Transact-SQL queries on DMVs and use the AlwaysOn Dashboard in SQL Server Management Studio to view the node votes and quorum model of the Windows cluster that hosts the availability group.

To view the quorum model of the Windows cluster hosting the availability group, query the DMV [sys.dm\_hadr\_cluster](http://technet.microsoft.com/library/hh212952(v=sql.110).aspx).

SELECT cluster\_name, quorum\_type\_desc, quorum\_state\_desc

FROM sys.dm\_hadr\_cluster;

When this query is run in the example environment discussed in this white paper, it returns the following.

cluster\_name quorum\_type\_desc quorum\_state\_desc

------------ ---------------- -----------------

EMU-AGClstr NODE\_AND\_FILE\_SHARE\_MAJORITY NORMAL\_QUORUM

To view the node votes, query the DMV [sys.dm\_hadr\_cluster\_members](http://technet.microsoft.com/library/hh231519(v=sql.110).aspx).

SELECT member\_name, number\_of\_quorum\_votes

FROM sys.dm\_hadr\_cluster\_members;

When this query is run in the example environment discussed in this white paper, it returns the following. (Vote allocation is covered in a later section.)

Member\_name number\_of\_quorum\_votes

----------- ----------------------

EMU-SQL1 1

EMU-SQL2 1

EMU-SQL3 0

FSWitness 1

You can also use the AlwaysOn Dashboard in SQL Server Management Studio to display quorum votes and the cluster state. Figure 5 shows this information for a Windows cluster with the Node Majority quorum model (cluster state and quorum votes are highlighted).

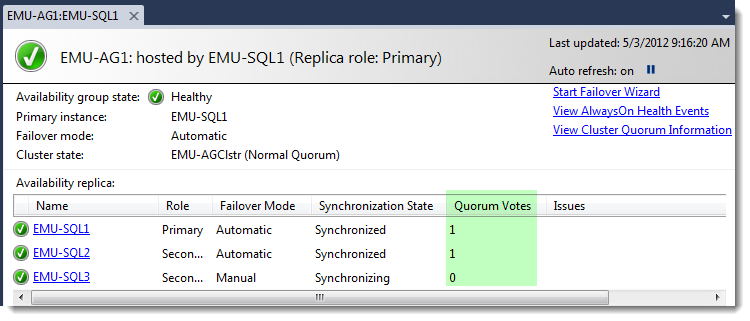


Figure : Displaying quorum votes and cluster state in the AlwaysOn Dashboard for the Node Majority quorum model

Although the **Quorum Votes** column is not displayed by default, you can add them it to the dashboard by right-clicking the **Availability replica** table column header and then selecting the specific column you want to display.

For a Node and File Share Majority quorum model, this AlwaysOn dashboard view shows only the nodes, not the file share. To see the complete quorum information, click **View Cluster Quorum Information** on the right. A pop-up window similar to Figure 6 appears.

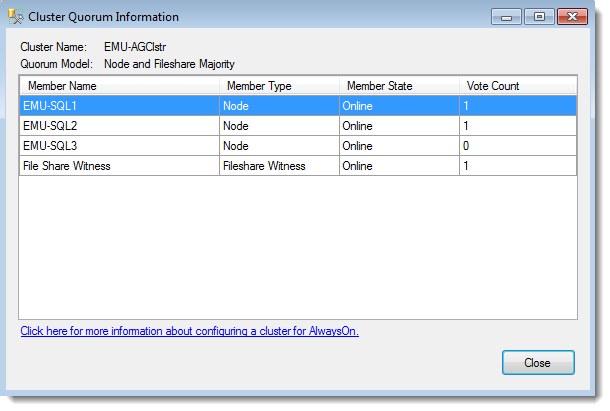


Figure : Cluster quorum information for Node and File Share Majority quorum model

### Configuring Node Votes

The NodeWeight property of the WSFC node represents the vote for that particular node. The following examples demonstrate how to configure the node NodeWeight from a node in a WSFC using Windows PowerShell. For executing Windows PowerShell on the server node, click **Start**, point to **Administrative Tools**, and then click **Windows PowerShell Modules**. In this example, EMU-SQL3 represents a specific WSFC node located in the secondary data center.

**To view current vote settings for all nodes**

Get-ClusterNode | fl NodeName, NodeWeight

**To set a node’s vote to “0”**

(Get-ClusterNode "EMU-SQL3").NodeWeight=0

**Note**: the value ‘0’ means the node doesn’t have a vote. The value ‘1’ means the node has a quorum vote.

## Client Connectivity

This section briefly discusses client connectivity considerations when moving to an availability group solution. For more information about client connectivity and application failover considerations, see [Client Connectivity and Application Failover (AlwaysOn Availability Groups)](http://msdn.microsoft.com/library/hh213417(v=sql.110).aspx).

### Legacy Database Mirroring Connection Strings

If you are migrating your application connections from a legacy database mirroring solution that designates the Failover Partner attribute, you can continue to use your database mirroring connection string only if the availability group is configured with a *single secondary replica*. The secondary replica cannot be enabled for read-only activity. You can designate the initial partner server name and optionally the failover partner name. Keep in mind that this is not a recommended long-term solution and that this workaround does not apply to the deployment solution that this white paper focuses on, because this specific design solution calls for a minimum of three replicas (one primary replica and two secondary replicas).

### Availability Group Listener

For availability groups, you can designate an availability group listener name for the server attribute. The availability group listener is a virtual network name (VNN) that you create for use with a specific availability group. This name is bound to one or more TCP/IP addresses and listener ports and is used to automatically connect to the primary replica, wherever it may be hosted at the time. The VNN eliminates the need to specify a failover partner attribute and allows for a scaled topology of up to five availability replica locations. For example, if an availability group fails over to Node2 from Node1, new connections to the availability group listener automatically connect to the replica currently hosting the primary replica.

Additionally, the availability group listener can also be used for automatic routing of read-only activity to read-only secondary replicas. For more information about this functionality, see [Configure Read-Only Routing for an Availability Group (SQL Server)](http://msdn.microsoft.com/library/hh710054.aspx).

### Multi-Subnet Connection Support

Regarding other availability group- related connection attributes, we recommend that availability group connection strings specify the MultiSubnetFailover attribute for both single and multi-subnet topologies when they reference an availability group listener name. When this connection string attribute is enabled, the MultiSubnetFailover connection option enables support for multi-subnet connections and opens up TCP sockets for the availability group listener IP addresses *in parallel*. If you are using older client libraries that do not support the MultiSubnetFailover attribute, you should consider increasing the client login timeout in the application connection string in order to account for potential multi-subnet connectivity latency. The timeout setting should reflect a value greater than the average availability group failover time per the specific environment. For more information about client support, see [SQL Server Native Client Support for High Availability, Disaster Recovery](http://msdn.microsoft.com/library/gg471494.aspx).

# Building the Availability Group Solution

This section discusses the steps and workflow required to create an availability group for a local high availability and remote disaster recovery solution. This paper focuses on creating a *new* environment similar to the topology shown in Figure 4 earlier. Remember that this specific design pattern assumes that *non-shared storage* is used for each SQL Server instance.

The minimum requirements for SQL Server 2012 are Windows Server 2008 R2 with Service Pack 1 (SP1) or Windows Server 2008 with SP2. The following instructions assume that Windows Server 2008 R2 SP1 Enterprise is the server node operating system.

Table 1 describes the steps required to build an availability group solution for local high availability and remote disaster recovery. Although each granular step is not rehashed here, the goal of this section is to help clarify the workflow sequence across the numerous implementation steps and participating job roles. Supporting documentation is referenced where appropriate. The following steps are broken out by job role because most large enterprise environments have a separation of duties across the database administrator, the Windows Server (or cluster) administrator, and the network administrator roles. It is important to properly communicate and coordinate activities across the roles.

| Step | Database administrator | Windows Server \ cluster administrator | Network administrator |
| --- | --- | --- | --- |
| 1. Begin the process by adding the Failover Clustering feature to the two newly configured nodes located in the primary data center and the third newly configured node located in the secondary data center. For more information about this process, see [Install the Failover Clustering Feature](http://technet.microsoft.com/library/cc770506.aspx) and [Understanding Requirements for Failover Clusters](http://technet.microsoft.com/library/cc771404.aspx). | Yes—for coordination of activities across roles | Yes |  |
| 1. Validate that the account you will be using to install and configure the WSFC is a domain account. This account should also have administrator permissions on each cluster node and **Create Computer Objects** and **Read All Properties** permissions for the container used for the domain computer accounts.   Alternatively, you can also prestage the Active Directory name object accounts ahead of time or use a domain administrator account for the installation. For more information about required permissions and provisioning options, including detailed instructions, see [Failover Cluster Step-by-Step Guide: Configuring Accounts in Active Directory](http://technet.microsoft.com/library/cc731002(WS.10).aspx). |  | Yes |  |
| 1. Using Failover Cluster Manager, perform cluster validation of the three server nodes across the two data centers that will be joining the WSFC. Because the design pattern we are discussing in this paper does *not* using shared storage, the shared storage tests are not performed during the validation.   After you run the cluster validation, ensure that there were no blocking issues identified before you create the actual WSFC. Even if you are permitted to continue to the next step with the existing warnings, it is important to investigate further in order to ensure a stable configuration. For more information, including instructions for performing a validation test, see [Validating a Failover Cluster Configuration](http://technet.microsoft.com/library/cc772055.aspx). |  | Yes | Yes–for any issues that may arise for the networking of the nodes |
| 1. After finishing validation, use Failover Cluster Manager to create a three-node WSFC. For more information about this process, see [Create a New Failover Cluster](http://technet.microsoft.com/library/cc755129.aspx).   Assuming that there are three nodes in your WSFC, your default quorum mode configuration will be **Node Majority**, at this point. You can change the quorum model later to Node and File Share Majority, if desired, or you can add another node as a voting-only node, without SQL Server installed. |  | Yes | Yes–for any issues that may arise for the networking of the nodes |
| 1. To prevent the disaster recovery data center node from affecting the availability of the primary data center nodes, install the hotfix detailed in KB 2494036, [A hotfix is available to let you configure a cluster node that does not have quorum votes in Windows Server 2008 and in Windows Server 2008 R2](http://support.microsoft.com/kb/2494036).   After you install this hotfix on each WSFC node, follow the steps detailed in the Configuring Quorum Node Votes section of this paper, setting the NodeWeight of the disaster recovery data center WSFC node to a 0 (zero) weight. This means that only the two nodes in the primary data center and the file share witness, which you will configure in the next step, will then have votes.  This workflow assumes you chose a file share witness instead of an additional server for node majority. If you had chosen an additional server to provide a vote instead, similar quorum vote considerations would still apply. |  | Yes |  |
| 1. Because the third node is in a separate data center and no longer has a vote, you should change this quorum model to Node and File Share Majority. Create a file share in the primary data center on a server node that will *not* be participating in the WSFC. This file share will act as the file share witness. After you create the file share, follow the instructions described earlier in this paper to change the quorum configuration to Node and File Share Majority.   Before you change the configuration, be sure that you have granted read and write permissions on the witness file share to the WSFC cluster account. |  | Yes |  |
| 1. Install a stand-alone instance of SQL Server 2012 Enterprise on each of the three WSFC nodes. Each node should have access to its own local, non-shared storage for use by SQL Server.   For each WSFC node, install the SQL Server 2012 Enterprise database engine (along with other optional features used in your environment), following the same steps you would follow to install a stand-alone SQL Server instance. Although this will be a typical stand-alone process, you should make sure that all of the SQL Server instances you install use the same SQL Server collation in order to host the availability group replicas (and furthermore, that they match the collation of your existing database mirroring and log shipping environment).  We also recommend that you use identical file paths on each node. | Yes |  |  |
| 1. Enable AlwaysOn Availability Group capabilities *for each SQL Server service*. For more information, including detailed steps for using SQL Server Configuration Manager or Windows PowerShell, see [Enable and Disable AlwaysOn Availability Groups](http://msdn.microsoft.com/library/ff878259(v=sql.110).aspx). | Yes |  |  |
| 1. After enabling all three SQL Server instances to support AlwaysOn Availability Groups, and after ensuring that the databases that will belong to the availability group are configured with FULL recovery model, back up your production user databases from the legacy topology and then restore them to a primary data center node in the WSFC.   It is assumed that you will be migrating one or more user databases to one SQL Server instance in the primary data center. The second node in the primary data center will then be used as a synchronous mode secondary replica.  You must also script out other SQL Server objects from the legacy topology that your user databases will depend on, but that are not contained within the restored user databases (such as SQL Server logins, associated server-level permissions, and SQL Server Agent jobs). This is similar to the process you would follow when you script dependent objects that are external to the mirrored database for a database mirroring partnership. There are several methods that can be used to transfer database objects between SQL Server instances. The Integration Services [Transfer SQL Server Objects task](http://technet.microsoft.com/library/ms142159(v=sql.110).aspx) is one such method. | Yes |  |  |
| 1. Create an availability group by using the SQL Server Management Studio Wizard ([Use the New Availability Group Wizard](http://msdn.microsoft.com/library/hh403415(v=sql.110).aspx)), Transact-SQL, or Windows PowerShell.  For more information about how to use Transact-SQL or SQL Server PowerShell, see [Create an Availability Group (Transact-SQL)](http://msdn.microsoft.com/library/ff878307(v=sql.110).aspx) or [Create an Availability Group (SQL Server PowerShell)](http://msdn.microsoft.com/library/gg492181(v=sql.110).aspx). | Yes |  | Yes–to ensure that the listener port that you designate for the availability group endpoint is open for each of the participating SQL Server instances |
| 1. Create the availability group listener (unless you have already created this as part of the previous step). You can create the availability group listener using a SQL Server Management Studio Wizard, Transact-SQL, or SQL Server PowerShell. For more information about the various methods, see [Create or Configure an Availability Group Listener (SQL Server)](http://msdn.microsoft.com/library/hh213080(v=sql.110).aspx). | Yes | Yes—to coordinate any firewall settings for the selected IPs | Yes–to coordinate IP address and port considerations |

Table : Building the Availability Groups solution by job role

For reference during the availability group installation, see Table 2 for a description of replica configurations that apply to the specific solution design that is the focus of this paper. Keep in mind that you can also choose to use the secondaries as readable secondaries. This is a viable choice that does not impact the overall design solution for HA and DR, and so it was not included in the table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data center | Replica | Role | Availability mode | Failover mode |
| Primary data center | Node 1 | Primary | Synchronous commit | Automatic |
| Primary data center | Node 2 | Secondary | Synchronous commit | Automatic |
| Disaster recovery data center | Node 3 | Secondary | Asynchronous commit (but a secondary synchronous replica is permitted; consider the network latency between the data centers, and its effect on performance to the application) | Manual |

Table : Replica settings

After you have completed the steps listed in Table 1, in Failover Cluster Manager you can see that a new resource group was created for the availability group. Within that resource group you can also find the availability group listener resource, associated listener IP addresses, and the availability group resource. Figure 7 shows an example of how this may look in Failover Cluster Manager.

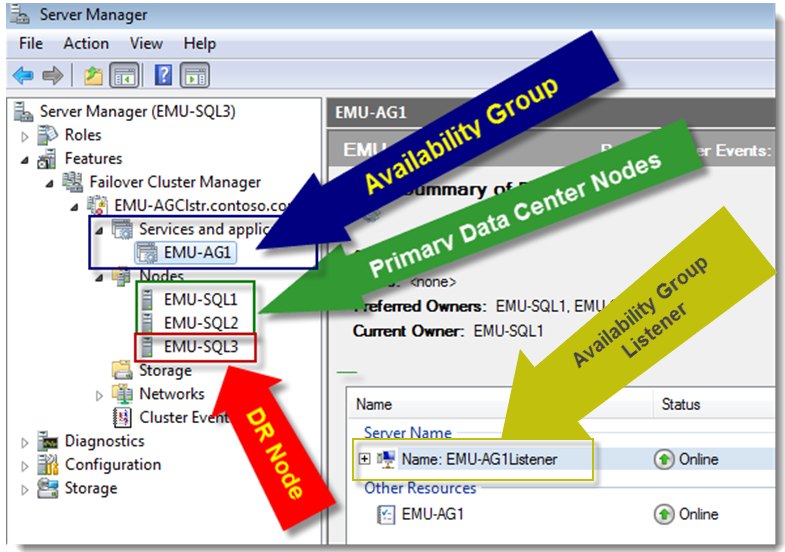


Figure : Windows Server Failover Cluster Manager: availability group for HA and DR solution

# Monitoring Considerations

* When you migrate from a database mirroring and log shipping topology to an availability group design solution topology, you will need to change your approach to monitoring the topology. The available methods and tools you can use for monitoring the availability group infrastructure include the following: AlwaysOn Group Dashboard in SQL Server Management Studio
* Object Explorer state information
* New availability-group related performance counters
* Catalog views
* Dynamic management views (DMVs)
* An Extended Events session that tracks recent AlwaysOn DDL-related statement executions, WSFC connectivity issues, failover events, state changes, and redo-thread blocking events

The AlwaysOn Group Dashboard is an efficient way to quickly identify the health of a specific availability group. From the dashboard you can identify the location of the primary instance, the failover mode of the replicas, the synchronization state of the replicas, and failover readiness of the various replicas (in other words, the risk of data loss for a specific replica). You can also open the AlwaysOn Health Events Extended Events session data directly from the dashboard in order to view recent availability group activity, state changes, and events.

You can also create SQL Server Agent alerts and job responses based on performance counter thresholds and availability group state changes. For more information about how to monitor an availability group environment, see [Monitoring of Availability Groups](http://msdn.microsoft.com/library/ff877954(v=sql.110).aspx).

# Recovering from a Disaster

This section details the workflow you should follow in the event of an outage of the WSFC nodes in the primary data center. This specific scenario assumes that the nodes in the primary data center are unavailable. This scenario also assumes that the only accessible node in the WSFC is in the secondary disaster recovery data center. Please note that a real-life disaster can involve a variety of outages. The guidance here is specific to a scenario where the primary data center nodes are unavailable. As mentioned earlier, it is also assumed that the remaining node does *not* (initially) have a quorum vote and that the only nodes with votes were located in the primary data center (Figure 4).

To recover an availability group in the event of a primary data center outage, follow these steps:

1. Failover Cluster Manager launched on the disaster recovery node is unlikely to initially provide useful information on the state of the WSFC because the cluster no longer has quorum. Additionally, the AlwaysOn Group Dashboard on the disaster recovery node is likely to show the failover mode as “unknown” and the availability replicas in a “resolving state” (only the local replica state is likely to be displayed during an outage). The availability databases may not be visible in the SQL Server Management Studio Object Explorer tree view on the disaster recovery node (see Figure 8 for an example view).

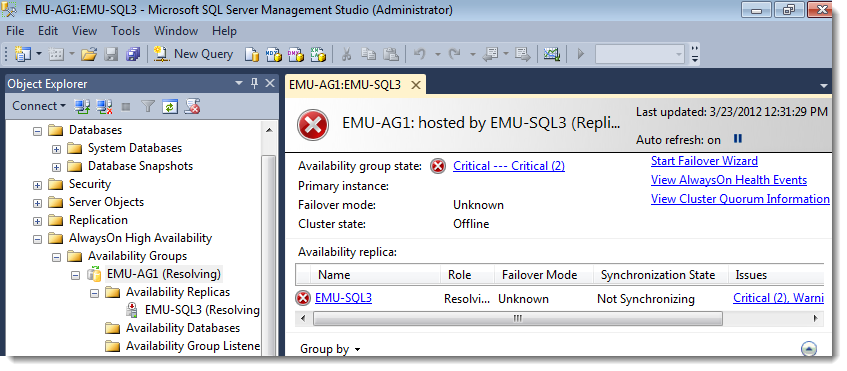


Figure : At the second data center during a primary data center outage

1. For a scenario where the primary data center’s status is uncertain and service must be restored from the disaster recovery secondary data center in order to conform to business recovery-time objectives, the only option is to start the Windows cluster after forcing quorum on the DR data center node. Force quorum should be used as a last resort, when the primary data center is expected to be down for extended period of time. When you force quorum, ensure that the nodes in the primary data center do not form their own quorum.

The following Windows PowerShell example illustrates forcing quorum on the node in the DR data center. First ensure that the cluster service is not already running on the DR node.

Stop-ClusterNode –Name "EMU-SQL3"

Then, start the cluster service by forcing quorum.

Start-ClusterNode –Name "EMU-SQL3" –FixQuorum

For more information about forcing quorum, see [Force a WSFC Cluster to Start Without a Quorum](http://msdn.microsoft.com/library/hh270275(v=sql.110).aspx).

1. At this time the quorum model of the cluster is still Node and File Share Majority.

Get-ClusterQuorum

Cluster QuorumResource QuorumType

------- -------------- ----------

EMU-AGClstr File Share Witness NodeAndFileShareMajority

Because the cluster is running on the DR node in a forced quorum state, you must adjust the quorum model and the node votes appropriately. Because there is only one node in the DR data center (in this topology example), change the quorum model to Node Majority (in this case, majority of one node) and assign one vote to the DR node and zero votes to the primary data center nodes. To set the quorum model to Node Majority, type the following command.

Set-ClusterQuorum -NodeMajority

Changing the quorum model changes the node votes to the default state (one vote to each node). Now adjust the node votes.

(Get-ClusterNode "EMU-SQL3").NodeWeight=1

(Get-ClusterNode "EMU-SQL1").NodeWeight=0

(Get-ClusterNode "EMU-SQL2").NodeWeight=0

At this point, the environment has one node and one vote, and thus one single point of failure.

Before you continue, validate that the node votes were modified as you intended by using the following Windows PowerShell command.

Get-ClusterNode | fl NodeName, NodeWeight

If you modified the WSFC “Cluster Group” owners to be constrained to the primary data center nodes only, you should also modify WSFC “Cluster Group” owners to include the disaster recovery node instead*.*

1. Bring the availability group online on the disaster recovery node SQL Server instance.

**Caution**: If the replica was configured with asynchronous mode, *restoring service could result in data loss for any unsent log records*. Be sure to fully understand the consequences of this action before you restore service. For more information about the risks of restoring service to replicas that were configured with asynchronous mode, see [Perform a Forced Manual Failover of an Availability Group](http://msdn.microsoft.com/library/ff877957(v=sql.110).aspx).

If the risk of some data loss is outweighed by the recovery-time objective (RTO) and the need to restore service at the data center, execute the following Transact-SQL syntax at on the disaster recovery SQL Server instance to force failover (in this example, EMU-AG1 is the name of the availability group).

ALTER AVAILABILITY GROUP [EMU-AG1] FORCE\_FAILOVER\_ALLOW\_DATA\_LOSS;

At this point, your databases within the availability group should now be available. You should double-check that any incoming application connections are disconnected from the *former* primary replica or are entirely disconnected. New connections to the availability group listener (which should be up again) should route automatically to the disaster recovery instance. You should also validate that application connections are no longer attempting to connect to the primary data center—avoiding a “split-brain” situation.

Also note that even after you re-establish quorum at the disaster recovery data center you will still see various warning messages about the primary data-center nodes being unavailable in SQL Server Management Studio. Figure 9 shows an example of what this may look like, including a view of Object Explorer and the AlwaysOn Group Dashboard.

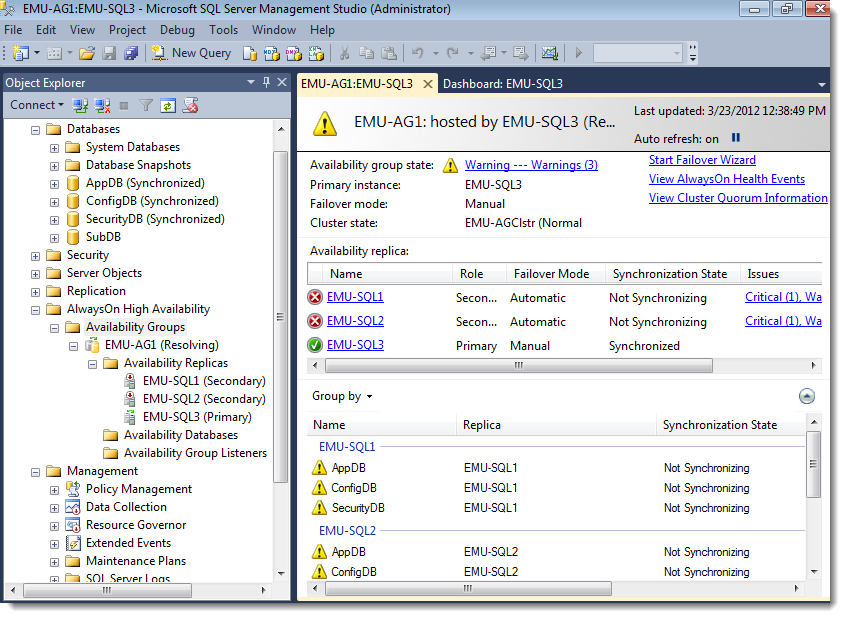


Figure : SQL Server Management Studio after a forced failover

As mentioned earlier in the paper, large enterprise environments typically have a separation of duties among the database administrator, Windows Server (or cluster) administrator, and network administrator roles. Table 3 recaps the previously described disaster recovery workflow, indicating which areas typically fall under the various enterprise roles from a planning perspective.

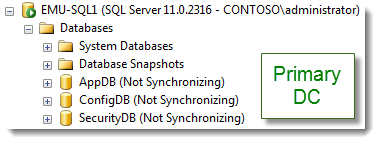
|  |  |  |  |
| --- | --- | --- | --- |
| Step | Database administrator | Windows Server \ cluster administrator | Network administrator |
| Confirm the current state of the primary data center and the remaining WSFC disaster recovery node, coordinating efforts. | Yes | Yes | Yes |
| Force quorum service on the disaster recovery node. |  | Yes |  |
| Remove votes from the primary nodes and add a vote to the DR node. |  | Yes |  |
| Force failover of the availability group to the disaster recovery SQL Server instance. | Yes |  |  |

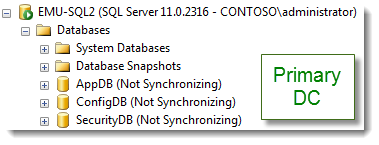
Table : Recovering from a disaster by job role

# Reverting to the Primary Data Center

The scenario here assumes that the restored service at the DR site is only for running the workload during a disaster, and it is desired to move the production workload back to the primary site soon after it is re-enabled to serve the workload. An outage scenario can have several variations and thus variations on recovery. The scenario described here assumes a disaster scenario where the primary data center servers were down for an extended period of time. After the issues with the primary data center are resolved, and the nodes in the primary data center are powered on again, the nodes attempt to connect to the WSFC. After they reconnect to the WSFC with cluster services running, the node weights assigned at the disaster recovery node should be in effect. Further, the scenario also assumes that the original SQL Server installs and associated databases are still intact.

At this point you should decide whether to salvage any data (that is, the data changes that were made in the original primary replica, but were not sent to the DR replica just prior to the disaster), or move forward with re-establishing all replica sessions. The replicas on the failed nodes will be in a “not synchronizing” state after a forced failover and the disaster recovery replica in a “synchronized” state as shown in Figure 10.





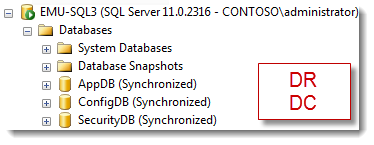


Figure Database state across replicas before the system reverts to the primary data center

One method to salvage the data from the original primary replica is to create a database snapshot on the suspended secondary database (that is, the original primary) for the purpose of extracting the appropriate data needed in order to resynchronize with the DR replica version of the availability databases. The following example demonstrates how to create a database snapshot on a “not synchronizing” availability database.

-- Create the database snapshot

CREATE DATABASE AppDB\_A1 ON

(NAME = AppDB, FILENAME =

'S:\Data\AppDB\_A1.ss' )

AS SNAPSHOT OF AppDB;

GO

Because the original scenario used asynchronous mode for the disaster recovery database, it is assumed that the recovery-point objective (RPO) tolerates some data loss. The next set of steps assumes that service will be restored using the existing primary data center replicas:

1. Start the controlled migration back to the primary data center by changing the quorum model appropriately (in this case, switch to Node and File Share Majority), and then readjust the votes.
2. After ensuring that the primary data center SQL server instances have started, on each SQL Server instance in the primary data center, execute the following Transact-SQL from the context of the **master** database to resume each database that is participating in the availability group.

USE [master]

GO

ALTER DATABASE AppDB SET HADR RESUME;

GO

ALTER DATABASE ConfigDB SET HADR RESUME;

GO

ALTER DATABASE SecurityDB SET HADR RESUME;

GO

1. Modify the availability group to temporarily use the synchronous commit availability mode, in order to synchronize prior to failover. The Transact-SQL command is as follows (executed on the current primary replica in the disaster recovery data center, where EMU-AG1 is our example availability group and EMU-SQL3 is the disaster recovery data center replica). Ideally the synchronous commit setting should be made during a period of low application activity in order to minimize the impact of transaction latency on users.

USE [master]

GO

ALTER AVAILABILITY GROUP [EMU-AG1]

MODIFY REPLICA ON N'EMU-SQL3' WITH (AVAILABILITY\_MODE = SYNCHRONOUS\_COMMIT);

GO

1. Confirm the synchronization status between the two locations (all the replica states should say “healthy” before moving to the next step, meaning that the replicas are synchronized).

SELECT role\_desc,

synchronization\_health\_desc

FROM sys.dm\_hadr\_availability\_replica\_states;

1. Fail over the AG from the disaster recovery data center node to the primary data center node (that is, connect and execute the following script on the primary data center node which will become the new primary replica).

ALTER AVAILABILITY GROUP [EMU-AG1] FAILOVER;

1. To match the original deployment,, change the disaster recovery replica node back to asynchronous commit. Execute the following Transact-SQL on the new primary replica, where EMU-SQL3 is the name of the disaster recovery replica and EMU-AG1 is the name of the availability group.

USE [master]

GO

ALTER AVAILABILITY GROUP [EMU-AG1]

MODIFY REPLICA ON N'EMU-SQL3' WITH

(AVAILABILITY\_MODE = ASYNCHRONOUS\_COMMIT);

GO

1. Remove the quorum vote from the WSFC node in disaster recovery data center.

The following table recaps the previously described disaster-recovery workflow, indicating which areas typically fall under the various enterprise roles from a planning perspective.

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Database administrator | Windows Server \ cluster administrator | Network administrator |
| 1. After you restore the primary data center service, change the quorum model appropriately. Then add back quorum votes to the primary data center nodes. |  | Yes |  |
| 1. Resume the availability database sessions on each secondary replica. | Yes |  |  |
| 1. Change the disaster recovery replica to synchronous commit. | Yes |  |  |
| 1. Confirm the synchronization status between the two locations (all replica states should say “healthy” before moving to the next step). | Yes |  |  |
| 1. Fail over to a replica in the primary data center. | Yes |  |  |
| 1. Revert the disaster recovery replica to asynchronous commit (to match original configuration). | Yes |  |  |
| 1. Remove the quorum vote from the node in the DR data center. |  | Yes |  |

Table : Reverting to the primary data center

# Conclusion

SQL Server 2012 AlwaysOn provides multiple options for building high availability (HA) and disaster recovery (DR) solution for your application. This white paper describes a solution that uses availability groups for HA and DR. This solution is purely a non-shared-storage solution, because each instance of SQL Server in the topology has its own copy of data, and does not need to share storage. You can use this solution to replace legacy topologies that use database mirroring and log shipping.

Successful deployment of such an HA/DR solution involves not just the DBA team, but close collaboration between the DBA team, Windows Server administration team, and the networking team in the IT organization. Cross-education of skills is very valuable when you deploy the HA/DR solution.

# References

* SQL Server 2012 AlwaysOn High Availability and Disaster Recovery Design Patterns (<http://go.microsoft.com/fwlink/?LinkId=255048>)
* Microsoft SQL Server AlwaysOn Solutions Guide for High Availability and Disaster Recovery (<http://msdn.microsoft.com/library/hh781257.aspx>)
* Overview of AlwaysOn Availability Groups (<http://technet.microsoft.com/library/ff877884(v=SQL.110).aspx>)
* Prerequisites, Restrictions, and Recommendations for AlwaysOn Availability Groups (<http://technet.microsoft.com/library/ff878487(v=sql.110).aspx>)
* Failover Cluster Step-by-Step Guide: Configuring the Quorum in a Failover Cluster (<http://technet.microsoft.com/library/cc770620(v=WS.10).aspx>)
* Windows Server hotfix for quorum votes (<http://support.microsoft.com/kb/2494036>)
* Windows PowerShell (<http://technet.microsoft.com/library/bb978526>)
* Mapping Cluster.exe Commands to Windows PowerShell Cmdlets for Failover Clusters (<http://technet.microsoft.com/library/ee619744(v=WS.10).aspx>)
* Windows PowerShell Survival Guide (<http://social.technet.microsoft.com/wiki/contents/articles/183.windows-powershell-survival-guide-en-us.aspx>)
* Failover Cluster Cmdlets in Windows PowerShell (<http://technet.microsoft.com/library/ee461009.aspx>)
* SQL Server PowerShell (<http://msdn.microsoft.com/en-us/library/hh245198.aspx>)

# Appendix A: Availability Groups HA/DR Example using 3 Data Centers

The architecture described throughout this white paper is based on two data centers, which is a common deployment topology. However, occasionally, some customers make use of a third data center for deployment. In most cases, the primary reason for such a solution is to provide automatic failover of the availability group between the primary and disaster recovery data centers. One way to achieve this is to deploy the two replicas in these two data centers and a file share witness in the third data center as shown in Figure 11.



Figure : Availability Groups HA/DR solution using three data centers

**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:%20AlwaysOn%20Architecture%20Guide:%20Building%20a%20High%20Availability%20and%20Disaster%20Recovery%20Solution%20by%20Uusing%20AlwaysOn%20Availability%20Groups%20)