Microsoft SQL Server AlwaysOn Solutions Guide for High Availability and Disaster Recovery

LeRoy Tuttle, Jr.

Quick Guide

Microsoft

Microsoft SQL Server AlwaysOn Solutions Guide for High Availability and Disaster Recovery

LeRoy Tuttle, Jr.

Contributors: Lindsey Allen, Justin Erickson, Min He, Cephas Lin, Sanjay Mishra

Reviewers: Kevin Farlee, Shahryar G. Hashemi (Motricity), Allan Hirt (SQLHA), Alexei Khalyako, Wolfgang Kutschera (Bwin Party), Charles Matthews, AyadShammout (Caregroup), David P. Smith (ServiceU), Juergen Thomas, Benjamin Wright-Jones

Summary: This white paper discusses how to reduce planned and unplanned downtime, maximize application availability, and provide data protection using SQL Server 2012 AlwaysOn high availability and disaster recovery solutions.

A key goal of this paper is to establish a common context for related discussions between business stakeholders, technical decision makers, system architects, infrastructure engineers, and database administrators.

Category: Quick Guide
Applies to: SQL Server 2012
Source: White paper (link to source content)
E-book publication date: May 2012
32 pages

Microsoft

This page intentionally left blank

Contents

High Availability and Disaster Recovery Concepts

You can make the best selection of a database technology for a high availability and disaster recovery solution when all stakeholders have a shared understanding of the related business drivers, challenges, and objectives of planning, managing, and measuring RTO and RPO objectives.

Readers who are familiar with these concepts can move ahead to the *Overview: High Availability with Microsoft SQL Server 2012* section of this paper.

Describing High Availability

For a given software application or service, high availability is ultimately measured in terms of the enduser's experience and expectations. The tangible and perceived business impact of downtime may be expressed in terms of informationloss, propertydamage, decreased productivity, opportunity costs, contractual damages, or the loss of goodwill.

*The principal goal of a high availability solution is to minimize or mitigate the impact of downtime.*A sound strategy for this optimally balances business processes and Service Level Agreements (SLAs) with technical capabilities and infrastructure costs.

A platform is considered highly available per the agreement and expectations of customers and stakeholders.The availability of a system can be expressed as this calculation:

$$\frac{Actual\ uptime}{Expected\ uptime} \times 100\%$$

The resulting value is often expressed by industry in terms of the number of 9's that the solution provides;meant to convey an annual number of minutes of possible uptime, or conversely, minutes of downtime.

Number of 9's	Availability Percentage	Total Annual Downtime
2	99%	3 days, 15 hours
3	99.9%	8 hours, 45 minutes
4	99.99%	52 minutes, 34 seconds
5	99.999%	5 minutes, 15 seconds

Planned vs. Unplanned Downtime

System outages are either anticipated and planned for, or they are the result of an unplanned failure.Downtime need not be considered negatively if it is appropriately managed.There are two key types of foreseeable downtime:

- **Planned maintenance.** A time window is preannounced and coordinated for planned maintenance tasks such as software patching, hardware upgrades, password updates, offline re-indexing, data loading, or the rehearsal of disaster recovery procedures. Deliberate, well-managed operational proceduresshould minimize downtime and prevent any data loss. Planned maintenance activities

can be seen as investments needed to prevent or mitigate other potentially more severe unplanned outage scenarios.

- **Unplanned outage.**System-level, infrastructure, or process failures may occur that are unplanned or uncontrollable, or that are foreseeable, but considered either too unlikely to occur, or are considered to have an acceptable impact.A robust high availability solution detects these types of failures, automatically recovers from the outage, and then reestablishes fault tolerance.

When establishing SLAs for high availability, you should calculate separate key performance indicators(KPIs) for planned maintenance activities and unplanned downtime. This approach allows you to contrast your investment in planned maintenance activities against the benefit of avoiding unplanned downtime.

Degraded Availability

High availability should not be considered as an all-or-nothing proposition.As an alternative to a complete outage, it is often acceptable to the enduser for a system to be partially available, or to have limited functionality or degraded performance.These varying degrees of availability include:

- **Read-only and deferred operations.**During a maintenance window, or during a phased disaster recovery, data retrieval is still possible, but new workflows and background processing may be temporarily halted or queued.
- **Data latency and application responsiveness.**Due to a heavy workload, a processing backlog, or a partial platform failure, limited hardware resources may be over-committed or under-sized.User experience may suffer, but work may still get done in a less productive manner.
- **Partial, transient, or impending failures.**Robustness in the application logic or hardware stack that retries or self-corrects upon encountering an error.These types of issues may appear to the end user as data latency or poor application responsiveness.
- **Partial end-to-end failure.**Planned or unplanned outages may occur gracefully within vertical layers of the solution stack (infrastructure, platform, andapplication), or horizontally between different functional components. Users may experience partial success or degradation, depending upon the features or components that are affected.

The acceptability of these suboptimal scenarios should be considered as part of a spectrum of degraded availability leading up to a complete outage, and as intermediate steps in a phased disaster recovery.

Quantifying Downtime

When downtime does occur, either planned, or unplanned, the primary business goal is to bring the system back online and minimize data loss.Every minute of downtime has direct and indirect costs.With unplanned downtime, you must balance the time and effort needed to determine why the outage occurred, what the current system state is, and what steps are needed to recover from the outage.

At a predetermined point in any outage, you should make or seek the business decision to stop investigating the outage or performing maintenance tasks, recover from the outage by bringing the system back online, and if needed, reestablish fault tolerance.

Recovery Objectives

Data redundancy is a key component of a high availability database solution. Transactional activity on your primary SQL Server instance is synchronously or asynchronously applied to one or more secondary instances.When an outage occurs, transactions that were in flight may be rolled back, or they may be lost on the secondary instances due to delays in data propagation.

You can both measure the impact, and set recovery goals in terms how long it takes to get back in business, and how much time latency there is in the last transaction recovered:

- **Recovery Time Objective (RTO).**This is the duration of the outage.The initial goal is to get the system back online in at least a read-only capacity to facilitate investigation of the failure.However, the primary goal is to restore full service to the point that new transactions can take place.

- **Recovery Point Objective (RPO).**This is often referred to as a measure of acceptable data loss. It is the time gap or latency between the last committed data transaction before the failure and the most recent data recovered after the failure. The actual data loss can vary depending upon the workload on the system at the time of the failure, the type of failure, and the type of high availability solution used.

You should use RTO and RPO values as goals that indicate business tolerance for downtime and acceptable data loss, and as metrics for monitoring availability health.

Justifying ROI or Opportunity Cost

Thebusiness costs of downtime may be either financial or in the form of customer goodwill.These costs may accrue with time, or they may be incurred at a certain point in the outage window.In addition to projecting the cost of incurring an outage with a given recovery time and data recovery point, you can also calculate the business process and infrastructure investments needed to attain your RTO and RPO goals or to avoid the outage all together.These investment themes should include:

- **Avoidingdowntime.**Outage recovery costs are avoided all together if an outage doesn't occur in the first place.Investments include the cost of fault-tolerant and redundant hardware or infrastructure, distributing workloads across isolated points of failure, and planned downtime for preventive maintenance.

- **Automating recovery.**If a system failure occurs, you can greatly mitigate the impact of downtime on the customer experience through automatic and transparent recovery.

- **Resource utilization.**Secondary or standby infrastructure can sit idle, awaiting an outage. Italso can be leveraged for read-only workloads, or toimprove overall system performance by distributing workloads across all available hardware.

For given RTO and RPO goals, the needed availability and recovery investments, combined with the projected costs of downtime,can be expressed and justified as a function of time. During an actual outage, this allows you to make cost-based decisions based on the elapsed downtime.

Monitoring Availability Health

From an operational point of view, during an actual outage, you should not attempt to consider all relevant variables and calculate ROI or opportunity costs in real time.Instead, you should monitor data latency on your standby instances as a proxy for expected RPO.

In the event of an outage, you should also limit the initial time spent investigating the root cause during the outage, and instead focus on validating the health of your recovery environment, and then rely upon detailed system logs and secondary copies of data for subsequent forensic analysis.

Planning for Disaster Recovery

While high availability efforts entail what you do to prevent an outage, disaster recovery efforts address what is done to re-establish high availability after the outage.

As much as possible, disaster recovery procedures and responsibilities should be formulated before an actual outage occurs.Based upon active monitoring and alerts, the decision to initiate an automated or manual failover and recovery plan should be tied to pre-established RTO and RPO thresholds.The scope of a sound disaster recovery plan should include:

- **Granularity of failure and recovery.**Depending upon the location and type of failure, you can take corrective action at different levels; that is, data center, infrastructure, platform, application, or workload.
- **Investigative source material**.Baseline and recent monitoring history, system alerts, event logs, and diagnostic queriesshould all be readily accessible by appropriate parties.
- **Coordination of dependencies**.Within the application stack, and across stakeholders, what are the system and business dependencies?
- **Decision tree.**A predetermined, repeatable, validated decision tree that includes role responsibilities, fault triage, failover criteria in terms of RPO and RTO goals, and prescribed recovery steps.
- **Validation.**After taking steps to recover from the outage, what must be done to verify that the system has returned to normal operations?
- **Documentation.** Capture all of the above items in a set of documentation, with sufficient detail and clarity so that a third party team can execute the recovery plan with minimal assistance. This type of documentation is commonly referred as a ‘run book’ or a ‘cook book’.
- **Recovery rehearsals.**Regularly exercise the disaster recovery plan to establish baseline expectations for RTO goals, and consider regular rotation of hosting the primary production site on the primary and each of the disaster recovery sites.

Overview:High Availability with Microsoft SQL Server 2012

Achieving the required RPO and RTO goals involves ensuring continuous uptime of critical applications and protection of critical data from unplanned and planned downtime.SQL Server provides a set of features and capabilities that can help achieve those goals while keepingthe cost and complexity low.

Readers who have a high-level familiarity with the new AlwaysOn capabilities can move ahead to the deeper coverage in the *SQL Server AlwaysOn Layers of Protection*section of this paper.

SQL Server AlwaysOn

AlwaysOn is a new integrated, flexible, cost-efficient high availability and disaster recovery solution.It can provide data and hardware redundancy within and across datacenters, and improvesapplication failover time to increase the availability of your mission-critical applications.AlwaysOn provides flexibility in configuration and enables reuse of existing hardware investments.

An AlwaysOn solution can leverage two major SQL Server 2012 features for configuring availability at both the database and the instance level:

- **AlwaysOn Availability Groups**, new in SQL Server 2012, greatly enhance the capabilities of database mirroring and helps ensure availability of application databases, and they enable zero data loss through log-based data movement for data protection without shared disks.

  Availability groups provide an integrated set of options including automatic and manual failover of a logical group of databases, support for up to four secondary replicas, fast application failover, and automatic page repair.

- **AlwaysOn Failover Cluster Instances (FCIs)** enhance the SQL Server failover clustering feature and support multisite clustering across subnets, which enables cross-data-center failover of SQL Server instances. Faster and more predictable instance failover is another key benefit that enables faster application recovery.

Significantly Reduce Planned Downtime

The key reason for application downtime in any organization is planned downtime caused by operating system patching, hardware maintenance, and so on. This can constitute almost 80 percent of the outages in an IT environment.

SQL Server 2012 helps reduce planned downtime significantly by reducing patching requirements and enabling more online maintenance operations:

- **Windows Server Core.**SQL Server 2012 supports deployments on Windows Server Core, a minimal, streamlined deployment option for Windows Server 2008 and Windows Server 2008 R2. This operating system configuration can reduce planned downtime by minimizing operating system patching requirements by as much as 60 percent.

- **Online Operations.**Enhanced support for online operations like LOB re-indexing and adding columns with default values helps to reduce downtime during database maintenance operations.

- **Rolling Upgrade and Patching.**AlwaysOn features facilitate rolling upgrades and patching of instances, which helps significantly to reduce application downtime.

- **SQL Server on Hyper-V.**SQL Server instances hosted in the Hyper-V environment receive the additional benefit of Live Migration, which enables you to migrate virtual machines between hosts with zero downtime. Administrators can perform maintenance operations on the host without impacting applications.

Eliminate Idle Hardware and Improve Cost Efficiency and Performance

Typical high availability solutions involve deployment of costly, redundant, passive servers.AlwaysOn Availability Groups enable you to utilize secondary database replicas on otherwise passive or idle servers for read-only workloads such as SQL Server Reporting Services report queries or backup operations.The ability to simultaneously utilize both the primary and secondary database replicas helps improve performance of all workloads due to better resource balancing across your server hardware investments.

Easy Deployment and Management

Features such as the Configuration Wizard, support for the Windows PowerShell command-line interface, dashboards, dynamic management views (DMVs), policy-based management, and System Center integration help simplifydeployment and management of availability groups.

Contrasting RPO and RTO Capabilities

The business goals for Recovery Point Objective (RPO) and Recovery Time Objective (RTO)should be key drivers in selecting a SQL Server technology for your high availability and disaster recovery solution. Thistable offers a rough comparison of the type of results that those different solutions may achieve:

High Availability and Disaster Recovery SQL Server Solution	Potential Data Loss (RPO)	Potential Recovery Time (RTO)	Automatic Failover	Readable Secondaries[1]
AlwaysOn Availability Group- synchronous-commit	Zero	Seconds	Yes[4]	0 - 2
AlwaysOn Availability Group- asynchronous-commit	Seconds	Minutes	No	0 - 4
AlwaysOn Failover Cluster Instance	NA[5]	Seconds -to-minutes	Yes	NA
Database Mirroring[2] **-** High-safety (sync + witness)	Zero	Seconds	Yes	NA
Database Mirroring[2] **-** High-performance (async)	Seconds[6]	Minutes[6]	No	NA
Log Shipping	Minutes[6]	Minutes -to-hours[6]	No	Not during a restore
Backup, Copy, Restore[3]	Hours[6]	Hours -to-days[6]	No	Not during a restore

[1]An AlwaysOn Availability Group can have no more than a total of four secondary replicas, regardless of type.

[2]This feature will be removed in a future version of Microsoft SQL Server. Use AlwaysOn Availability Groups instead.

[3]Backup, Copy, Restore is appropriate for disaster recovery, but not for high availability.

[4] Automatic failover of an availability group is not supported to or from a failover cluster instance.

[5]The FCI itself doesn't provide data protection; data loss is dependent upon the storage system implementation.

[6] Highly dependent upon the workload, data volume, and failover procedures.

SQL Server AlwaysOn Layers of Protection

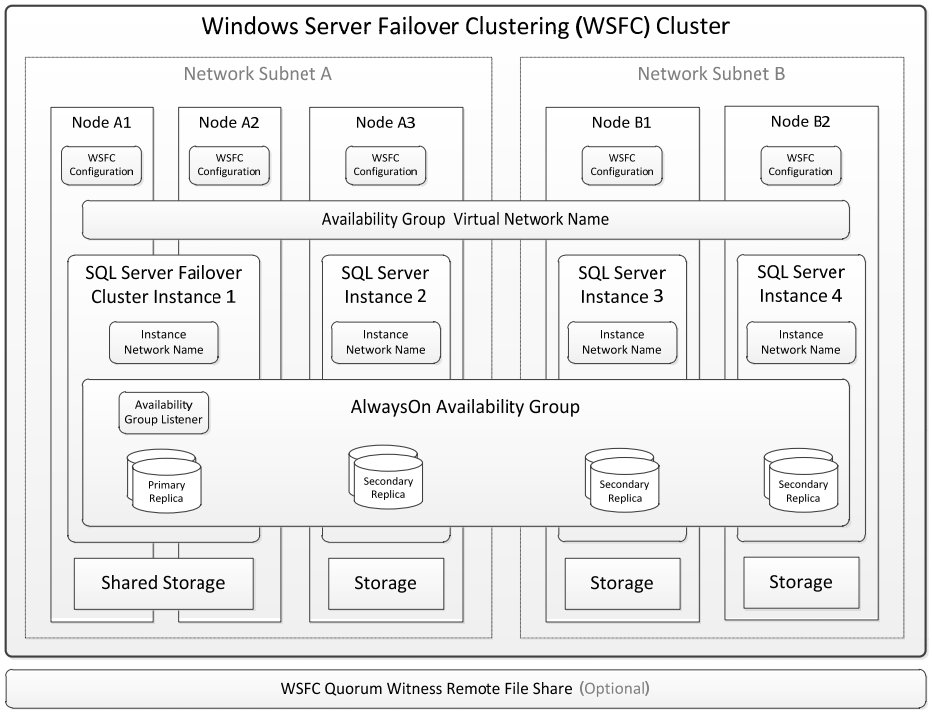
SQL Server AlwaysOn solutions help provide fault tolerance and disaster recovery across several logical and physical layers of infrastructure and application components. Historically, it has been a common practice to have a separation of duties and responsibilities for the various involved audiences and roles, such that each was predominately only concerned a portion of those solution layers.

This section of the paper is organized to walk through a deeper description of each of those layers, and to offer rationale and guidance for your design discussions and implementation decisions.

A successful SQL Server AlwaysOn solution requires understanding and collaboration across these layers:

- **Infrastructure level.**Server-level fault-tolerance and intra-nodenetwork communication leverages *Windows Server Failover Clustering* (WSFC) features for health monitoring and failover coordination.
- **SQL Server instance level.**A SQL Server AlwaysOn F*ailover Cluster Instance*(FCI) is a SQL Server instance that is installed across and can fail over to server nodes in a WSFC cluster.The nodes that host the FCIare attached to robust symmetric shared storage (SAN or SMB).
- **Database level.**An *availability group*is a set of user databases that fail over together. An availability group consists of a primary replica and one to four secondary replicas. Each replica is hosted by an instance of SQL Server (FCI or non-FCI) on a different node of the WSFC cluster.
- **Client connectivity.**Database client applications can connect directly to a SQL Server *instance network name*, or they may connect to a *virtual network name (VNN)* that is bound to an *availability group listener.*The VNN abstracts the WSFC cluster and availability group topology, logicallyredirectingconnection requests to the appropriate SQL Server instance and database replica.

The logical topology of a representative AlwaysOn solution is illustrated in this diagram:



Infrastructure Availability

Both AlwaysOn Availability Groups and AlwaysOn Failover Cluster Instances leverage the Windows Server operating system and WSFC as a platform technology.More than ever before, successful Microsoft SQL Server database administrators will rely upon a solid understanding of these technologies.

Windows Operating System

SQL Server relies upon the Windows platform to provide foundational infrastructure and services for networking, storage, security, patching, and monitoring.

The different editions of SQL Server 2012 progressively build upon the increasing capabilities and capacity of similar editions of the Windows Server 2008 R2 operating system, including Windows Server 2008 R2 Standard operating system, Windows Server 2008 R2Enterprise operating system, and Windows Server 2008 R2 Datacenteroperating system.

For more information, see:Hardware and Software Requirements for Installing SQL Server 2012(http://msdn.microsoft.com/en-us/library/ms143506(SQL.110).aspx).

Windows Server Core Installation Option

As a key high-availability feature, SQL Server 2012 supports deployment on the Server Core installation option in Windows Server 2008 or later.The *Server Core* installation option provides a minimal environment for running specific server roles with limited functionality and very limited GUI application support. By default, only necessary services and a command-prompt environment are enabled.

This mode of operation reducesthe operating system attack surface and system overhead, and it can significantly reduce ongoing maintenance, servicing, and patching requirements.

A key consideration for deploying SQL Server 2012 on Windows Server Core is that all deployment, configuration, administration, and maintenance of SQL Server and of the operating system must be done usinga scripting environment such as Windows PowerShell, or through the use of command-line or remote tools.

Optimizing SQL Server for Private Cloud

High availability and disaster recovery scenarios are increasingly critical in the Private Cloud environment. Deploy SQL Server to your Private Cloud to help ensure that your computer, network and storage resources are used efficiently, reducing both physical footprint and capital and operational expenses. It helps youconsolidate deployments, scale your resources efficiently, and deploy resources on demand without compromising control.

In addition to Windows Server Failover Clustering support for both Hyper-V host and guest systems, SQL Server also supports Live Migration, which is the ability to move virtual machines between hosts with no discernible downtime.Live Migration also works in conjunction with guest clustering.

For more information, seePrivate Cloud Computing - Optimizing SQL Server for Private Cloud(http://www.microsoft.com/SqlServerPrivateCloud).

Windows Server Failover Clustering

Windows Server Failover Clustering (WSFC) provides infrastructure features that support the high-availability and disaster-recovery scenarios of hosted server applications such as Microsoft SQL Server.

If a WSFC cluster node or service fails, the services or resources that were hosted on that node can be automatically or manually transferred to another available node in a process known as *failover*. With AlwaysOn solutions, this process applies to both FCIs and to availability groups.

The nodes in the WSFC cluster work together to collectively provide these types of capabilities:

- **Distributed metadata and notifications.**WSFC service and hosted application metadata is maintained on each node in the cluster. This metadata includes WSFC configuration and status in addition to hosted application settings. Changes to the metadata or status on one node are automatically propagated to the other nodes in the cluster.

- **Resource management.**Individual nodes in the cluster may provide physical resources such as direct-attached storage (DAS), network interfaces, and access to shared disk storage. Hosted applications,such as SQL Server, register themselves as a cluster resource, and they can configure startup and health dependencies upon other resources.

- **Health monitoring.** Internode and primary node health detection is accomplished through a combination of heartbeat-style network communications and resource monitoring. The overall health of the cluster is determined by the votes of a quorum of nodes in the cluster.

- **Failover coordination.**Each resource is configured to be hosted on a primary node, and each can be automatically or manually transferred to one or more secondary nodes. A health-based failover policy controls automatic transfer of resource ownership between nodes. Nodes and hosted applications are notified when failover occurs so that they can react appropriately.

For more information, seeWindows Server | Failover Clustering and Node Balancing(http://www.microsoft.com/windowsserver2008/en/us/failover-clustering-main.aspx).

Note: It is now critically important that database administrators understand the inner workings of WSFC clusters and quorum management. AlwaysOn health monitoring, management, and failure recovery steps are all intrinsically tied to your WSFC configuration.

WSFC Storage Configurations

Windows Server Failover Clustering relies upon each node in the cluster to manage its connected storage devices, disk volumes, and file system.WSFC assumes that the storage subsystem is extremely robust, and therefore if the storage device attached to a node is unavailable, the cluster node is considered to be at fault.

For write-based operations, a disk volume is logically attached to a single cluster node at a time using a SCSI-3 persistent reservation.Depending upon storage subsystem capabilities and configuration, if a node fails, logical ownership of the disk volume can be transferred to another node in the cluster.

SQL Server AlwaysOn solutions both leverage and are restricted to certain WSFC storage configuration combinations, including:

- **Direct-attached vs. remote.**Storage devices are directly physically attached to the server, or they are presented by a remote device through a network or *host bus adaptor* (HBA).Remote storage technologies include *Storage Area Network* (SAN) based solutions such as iSCSI or Fibre Channel, as well as *Server Messaging Block* (SMB) file share based solutions.

- **Symmetric vs. asymmetric.**Storage devices are considered symmetric if exactly the same logical disk volume configuration and file paths are presented to each node in the cluster. The physical implementation and capacity of the underlying disk volumes can vary.

- **Dedicated vs. shared.**Dedicated storage is reserved for use and assigned to a single node in the cluster.Shared storage is accessible to multiple nodes in the cluster. Control and ownership of compliant shared storage devices can be transferred from one node to another using SCSI-3 protocols.WSFC supports the concurrent multi-node hosting of *cluster shared volumes* for file sharing purposes.However, SQL Server does not support concurrent multi-node access to a shared volume.

Note:SQL Server FCIs still require symmetrical shared storage to be accessible by all possible node owners of the instance.However, with the introduction of AlwaysOn Availability Groups, you can now deploy different non-FCI instances of SQL Server in a WSFC cluster, each with its own unique, dedicated, local or remote storage.

WSFC Resource Health Detection and Failover

Each resource in a WSFC cluster node can report its status and health, periodically or on-demand.A variety of circumstances may indicate a cluster resource failure, including: power failure, disk or memory errors, network communication errors, misconfiguration, or nonresponsive services.

You can make WSFC cluster resources such as networks, storage, or services dependent upon one another. The cumulative health of a resource is determined by successive rollup of its health with the health of each of its resource dependencies.

For AlwaysOn Availability Groups, the availability group and the availability group listener are registered as WSFC cluster resources.For AlwaysOn Failover Cluster Instances, the SQL Server service and the SQL Server Agent service are registered as WSFC cluster resources, and both are made dependent upon the instance's virtual network name resource.

If a WSFC cluster resource experiences a set number of errors or failures over a period of time, the configured *failover policy* causes the cluster service to do one of the following:

- Restart the resource on the current node.
- Set the resource offline.
- Initiate an automatic failover of the resource and its dependencies to another node.

Note:WSFC cluster resource health detection has no direct impact on the individual node's health or the overall health of the cluster.

WSFC Cluster Validation Wizard

The cluster validation wizard is a feature that is integrated into failover clustering in Windows Server 2008 and Windows Server 2008 R2. It is a key toolfor a database administrator to use to help ensure that a clean, healthy, stable WSFC environment exists, *before*deploying a SQL Server AlwaysOn solution.

With the cluster validation wizard, you can run a set of focused tests on either a collection of servers that you intend to use as nodes in a cluster, or on an existing cluster. This process tests the underlying hardware and software directly, and individually, to obtain an accurate assessment of how well a WSFC clusterwouldbe supported on a given configuration.

This validation process consists of a series of tests and data collection on each node in these categories:

- **Inventory**.Information on BIOS versions, environment levels, host bust adapters, RAM, operating system versions, devices, services, drivers, and so on.
- **Network**.Information on NIC binding order, network communications, IP configuration, and firewall configuration.Validates inter-node communications on all NICs.
- **Storage**.Information on disks, drive capacity, access latency, files systems, and so on.Validates SCSI commands, disk failover functionality, and symmetric or asymmetric storage configuration.
- **System configuration**.Validates Active Directory configuration, that drivers are signed, memory dump settings, required operating system features and services, compatible processor architecture, and service pack and Windows Software Update levels.

The results of these validation tests give you information needed to fine-tune a cluster configuration, track the configuration, and identify potential cluster configuration issues before they cause downtime. You can save a report of the tests results as a HTML document for later reference.

You should run these tests before and after you make any changes to WSFC configuration, before you install SQL Server, and as a part of any disaster recovery process.A cluster validation report is required by Microsoft Customer Support Services (CSS) as a condition of Microsoft supporting a given WSFC cluster configuration.

For more information, seeFailover Cluster Step-by-Step Guide: Validating Hardware for a Failover Cluster (http://technet.microsoft.com/en-us/library/cc732035(WS.10).aspx).

Note:If your cluster configuration has asymmetric storage, as is the case with hardware-based geo-clustering storage solutions, or as may be the case with AlwaysOn Availability Groups, you may need to apply a number of hotfixes to prevent the cluster validation wizard from failingthe storage validation steps.

For more information, see Prerequisites, Restrictions, and Recommendations for AlwaysOn Availability Groups (http://msdn.microsoft.com/en-us/library/ff878487(SQL.110).aspx#SystemReqsForAOAG).

WSFC Quorum Modes and Voting Configuration

WSFC uses a quorum-based approach to monitoring overall cluster health and maximize node-level fault tolerance.A fundamental understanding of WSFC quorum modes and node voting configuration is very important to designing, operating, and troubleshooting your AlwaysOn high availability and disaster recovery solution.

Cluster Health Detection by Quorum

Each node in a WSFC cluster participates in periodic heartbeat communication to share the node's health status with the other nodes. Unresponsive nodes are considered to be in a failed state.

A *quorum*node set is a majority of the voting nodes and witnesses in the WSFC cluster.The overall health and status of a WSFC cluster is determined by a periodic *quorum vote*.The presence of a quorum means that the cluster is healthy enough to provide node-level fault tolerance.

The absence of a quorum indicates that the cluster is not healthy.Overall WSFC cluster health must be maintained in order to ensure that healthy secondary nodes are available for primary nodes to failover to.If the quorum vote fails, the entire WSFC clusteris set offline as a precautionary measure.This also causesallSQL Server instances registered with the cluster to be stopped.

Note: If a WSFC cluster is set offline because of quorum failure, manual intervention is required to bring it back online.For more information, see the *WSFC Disaster Recovery through Forced Quorum* section later in this paper.

Quorum Modes

A *quorum mode* is configured at the WSFC cluster level to specify the methodology used for quorum voting.The Failover Cluster Manager utilityrecommends a quorum mode based on the number of nodes in the cluster.

One of the following quorum modes determines what constitutes a quorum of votes:

- **Node Majority.**More thanone-halfof the voting nodes in the cluster must vote affirmatively for the cluster to be healthy.
- **Node and File Share Majority.**Similar to Node Majority quorum mode, except that a remote file share is also configured as a voting witness, and connectivity from any node to that share is also counted as an affirmative vote.More thanhalfof the possible votes must be affirmative for the cluster to be healthy.

  As a best practice, the witness file share should not reside on any node in the cluster, and it should be visible to all nodes in the cluster.
- **Node and Disk Majority.**Similar to Node Majority quorum mode, except that a shared disk cluster resource is also designated as a voting witness, and connectivity from any node to that shared disk is also counted as an affirmative vote.More than half of the possible votes must be affirmative for the cluster to be healthy.

- **Disk Only.**A shared disk cluster resource is designated as a witness, and connectivity by any node to that shared disk is counted as an affirmative vote.

For more information, seeFailover Cluster Step-by-Step Guide: Configuring the Quorum in a Cluster(http://technet.microsoft.com/en-us/library/cc770620(WS.10).aspx).

Note:Unless each node in the cluster is configured to use the same shared storage quorum witness disk, you should generally use the Node Majorityquorum mode if you have an odd number of voting nodes, or the Node and File Share Majorityquorum modeif you have an even number of voting nodes.

Voting and Non-Voting Nodes

By default, each node in the WSFC cluster is included as a member of the cluster quorum; each node, file share witness, and disk witnesshas a single vote in determining the overall cluster health.The quorum discussion to this point in this paper has carefully qualified the set of WSFC cluster nodes that vote on cluster health as *voting nodes*. In some circumstances, you may not want every node to have a vote.

Each node in a WSFC cluster continuously attempts to establish a quorum. No individual node in the cluster can definitively determine that the cluster as a whole is healthy or unhealthy.At any given moment, from the perspective of each node, some of the other nodes may appearto be offline, or appear to be in the process of failover, or appear unresponsive due to a network communication failure.A key function of the quorum vote is to determine whether the apparent state of each of node in the WSFC cluster is indeed that actual state of those nodes.

For all of the quorum models except Disk Only, the effectiveness of a quorum vote depends on reliable communications among all of the voting nodes in the cluster.You should trust the quorum vote when all nodes are on the same physical subnet.

However, if a node on another subnet is seen as nonresponsive in a quorum vote, but it is actually online and otherwise healthy, that is most likelydue to a network communications failure between subnets.Depending upon the cluster topology, quorum mode, and failover policy configuration, that network communications failure may effectively create more than one set (or subset) of voting nodes.

If more than one subset of voting nodes is able to establish a quorum on its own, that is known as a *split-brain scenario*.In such a scenario, the nodes in the separate quorums may behave differently, and in conflict with one another.

Note:The split-brain scenario is possible only if a system administrator manually performs a forced quorum operation, or in very rare circumstances, a forced manual failover, explicitly subdividing the quorum node set.For more information, see the *WSFC Disaster Recovery through Forced Quorum* section later in this paper.

To simplify your quorum configuration and increase up-time, you may want to adjust each node's *NodeWeight*setting (a value of 0 or 1) so that the node's vote is not counted towards the quorum.

Recommended Adjustments to Quorum Voting

To determine the recommended quorum voting configuration for the cluster, apply these guidelines, in sequential order:

1. **No vote by default.** Assume that each node should not vote without explicit justification.
2. **Include all primary nodes.**Each node that hosts an AlwaysOn Availability Group primary replica or is the preferred owner of the AlwaysOn Failover Cluster Instance should have a vote.
3. **Include possible automatic failover owners.**Each node that could host a primary replica or FCI, as the result of an automatic failover, should have a vote.
4. **Exclude secondary site nodes.**In general, do not give votes to nodes that reside at a secondary disaster recovery site.You do not want nodes in the secondary site to contribute to a decision to take the cluster offline when there is nothing wrong with the primary site.
5. **Odd number of votes.**If necessary, add a witness file share, a witness node (with or without a SQL Server instance), or a witness disk to the cluster and adjust the quorum mode to prevent possible ties in the quorum vote.
6. **Reassess vote assignments post-failover.**You do not want to fail over into a cluster configuration that does not support a healthy quorum.

For more information on adjusting node votes, see Configure Cluster Quorum NodeWeight Settings(http://msdn.microsoft.com/en-us/library/hh270281(SQL.110).aspx).

You cannot adjust the vote of a file share witness. Instead, you must select a different quorum mode to include or exclude its vote.

Note:SQL Server exposes several system dynamic management views (DMVs) that can help you administer settings related WSFC cluster configuration and node quorum voting.

For more information, seeMonitor Availability Groups(http://msdn.microsoft.com/en-us/library/ff878305(SQL.110).aspx).

WSFC Disaster Recoverythrough Forced Quorum

Quorum failure is usually caused by a systemic disaster or a persistent communications failure involving several nodes in the WSFC cluster.Remember that quorum failure causesall clustered services, SQL Server instances, and Availability Groups in the WSFC cluster to be set offline, because the cluster cannot ensure node-level fault tolerance.A quorum failure means that healthy voting nodes in the WSFC cluster no longer satisfy the quorum model.Some nodes may have failed completely, and some may have just shut down the WSFC service and are otherwise healthy, except for the loss of the ability to communicate with a quorum.

To bring the WSFC cluster back online, you must correct the root cause of the quorum failureon at least one node under the existing configuration. In a disaster scenario, you may need to reconfigure or identify alternative hardware to use. You may also want to reconfigure the remaining nodes in the WSFC cluster to reflect the surviving cluster topology as well.

You can use the *forced quorum* procedure on a WSFC cluster node to override the safety controls that took the cluster offline.This effectively tells the cluster to suspend the quorum voting checks, and lets you bring the WSFC cluster resources and SQL Server back online on any of the nodes in the cluster.

This type of disaster recovery process should include the following steps:

1) **Determine the scope of the failure.**Identify which availability groups or SQL Server instances are nonresponsive and which cluster nodes are online and available for post-disaster use, and then examine the Windows event logs and the SQL Server system logs.Where practical, you should preserve forensic data and system logs for later analysis.

2) **Start the WSFC cluster by using forced quorum on a single node.**On an otherwise healthy node, manually force the cluster to come online using the forced quorum procedure.To minimize potential data loss, select a node that was last hosting an availability group primary replica.

   For more information, seeForce a WSFC Cluster to Start Without a Quorum(http://msdn.microsoft.com/en-us/library/hh270275(v=SQL.110).aspx).

   Note:If you use theforced quorum setting,quorum checksare blocked cluster-wide until theWSFC cluster achieves a majority of votes and automatically transitions to a regular quorum mode of operation.

3) **Start the WSFC service normally on each otherwise healthynode, one at a time.**You do not have to specify the forced quorum option when you start the cluster service on the other nodes.

   As the WSFC service on each node comes back online, it negotiates with the other healthy nodes to synchronize the new cluster configuration state.Remember to do this one node at a time to prevent potential race conditions in resolving the last known state of the cluster.

   Note:Ensure that each node that you start cancommunicate with the other newly online nodes, or you run the risk of creating more than one quorum node set; that is a split-brain scenario. If your findings in step 1 are accurate, this should not occur.

4) **Apply new quorum mode and node voteconfiguration.**If you successfully restarted all nodes in the cluster using the forced quorum procedure, and if you corrected the root cause of the quorum failure, you do not need to make changes to the original quorum mode and node vote configuration.

   Otherwise, you should evaluate the newly recovered cluster node and availability replica topology, and change the quorum mode and vote assignments for each node as appropriate. Set the WSFC cluster service on unrecovered nodes offline, or set their node votes to zero.

   Note:At this point, the nodes and SQL Server instances in the cluster may appear to be restored back to regular operation.However, a healthy quorum may still not exist.Using Failover Cluster Manager, or the AlwaysOn Dashboard within SQL Server Management Studio, or the appropriate DMVs, verify that a healthy quorum has been restored.

5) **Recover availability group database replicasas needed.**Some databases may recover and come back online on their own as part of the regular SQL Server startup process.The recovery of other databases may require additional manual steps.

   You can minimize potential data loss and recovery time for the availability group replicas by bringing them back online in this sequence, if possible:primary replica, synchronous secondary replicas, asynchronous secondary replicas.

6) **Repair or replace failed components and revalidate the cluster.**Now that you have recovered from the initial disaster and quorum failure, you should repair or replace the failed nodes and adjust related WSFC and AlwaysOn configurations accordingly.This can include dropping availability group replicas, evicting nodes from the cluster, or flattening and reinstalling software on a node.

   Note:You must repair or remove all failed availability replicas.SQL Server 2012 doesnot truncate thetransaction log past the last known point of the farthest behind availability replica.If a failed replica is not repaired or removed from the availability group, the transaction logs will grow and you will run the risk of running out of transaction log space on the other replicas.

7) **Repeat step 4 as needed.**The goal is to reestablish the appropriate level of fault tolerance and high availability for healthy operations.

8) **Conduct RPO/RTO analysis.**You should analyze SQL Server system logs, database timestamps, and Windows event logs to determine root cause of the failure, and to document actual Recovery Point and Recovery Time experiences.

SQL Server Instance Level Protection

The next layer of protection in an AlwaysOn solution is the data platform itself; these are the capabilities and features offered by Microsoft SQL Server 2012 and its integration with Windows Server infrastructure components.

Availability Improvements – SQL Server Instances

These are new SQL Server 2012 instance-level features that enhance availability for both AlwaysOn Failover Cluster Instances, as well as for stand-alone instances that host AlwaysOn Availability Groups.

These improvements represent enhancements for managing and troubleshooting failover scenarios:

- **Flexible Failover Policy.**The output of the new system stored procedure used for robust failure detection, **sp_server_diagnostics**, uses the **FailureConditionLevel** property to convey the severity of a failure affecting the SQL Server instance.A WSFC failover policy governs how this value impacts the SQL Server instance; ranging from relative tolerance of errors, to being sensitive to any SQL Server internal component error.

  You can configure failover to be triggered by any one of a range of error levels, including:server down, server unresponsive, critical error, moderate error, or any qualified error.The **FailureConditionLevel** property can be used for FCI or availability group failover policies.

  Prior to SQL Server 2012, there was no granularity of error conditions to govern failover; any service-level failure caused failover.

  For more information, seeFailover Policy for Failover Cluster Instances (http://msdn.microsoft.com/en-us/library/ff878664(SQL.110).aspx).

- **Enhanced instrumentation and logging.**There are a number of AlwaysOn-specific system configuration views, DMVs, performance counters, and an extended event health session that captures and dumps information needed to troubleshoot, tune, and monitor your AlwaysOn deployment.Many of these are exposed via new SQL Server Policy Management facets and policies.

  For more information, see AlwaysOn Availability Groups Dynamic Management Views and Functions (http://msdn.microsoft.com/en-us/library/ff877943(SQL.110).aspx),and sys.dm_os_cluster_nodes (http://msdn.microsoft.com/en-us/library/ms187341(SQL.110).aspx).

- **SMB file share support.**You can place database files on a Windows Server 2008 or later remote file share for both stand-alone and failover cluster instances, negating the need for a separate drive letter per FCI.This is a good option for storage consolidation or for hosting database file storage on a physical server for a virtual machine guest operating system.With the right configuration, I/O performance can very nearly approximate that of direct-attached storage.

  For more information, seeSQL Databases on File Shares - It's time to reconsider the scenario(http://blogs.msdn.com/b/sqlserverstorageengine/archive/2011/10/18/sql-databases-on-file-shares-it-s-time-to-reconsider-the-scenario.aspx).

Note: In a WSFC cluster, you cannot add a SMB file share resource dependency to the SQL Server resource group; you must take separate measures to ensure the availability of the file share.If the file share becomes unavailable, SQL Server throws an I/O exception and goes offline.

- **WSFC interoperability with DNS.**The virtual network name (VNN) for an FCI or availability group listener is registered with DNS only during VNN creation or during configuration changes.Allvirtual IP addresses, regardless of online or offline state, are registered with DNS under the same virtual network name.Client calls to resolve the virtual network name in DNS return all of the registered IP address in a varying round-robin sequence.

AlwaysOn Failover Cluster Instances

The primary purpose of anAlwaysOn SQL Server Failover Cluster Instance (FCI) is to enhance availability of a SQL Server instance hosted on local server and storage hardware within a single data center.

An FCIis a single logical SQL Server instance that is installed across nodes in a Windows Server Failover Clustering (WSFC) cluster, but only active on one node at a time.Client applications connect to a virtual network name and virtual IP address that are owned by the active cluster node.

Each installed node has an identical configuration and set of SQL Server binaries.The WSFC cluster service also replicates relevant changes from the activeinstance'sentries in the Windows registry to each installed node.Each node that the FCI is installed on is designated as a possible owner of the instance and its resources, within a preferred failover sequence.

Database files are stored on shared symmetrical storage volumes are registered as a resource with the WSFC cluster, and are owned by the node thatcurrently hosts the FCI.

For more information, seeAlwaysOn Failover Cluster Instances(http://msdn.microsoft.com/en-us/library/ms189134(SQL.110).aspx).

FCI Failover Process

If a dependent cluster resource fails, an AlwaysOn Failover Cluster Instance interacts with the WSFC cluster service using this high-level process to do a failover:

1) **A restart isindicated.**A periodic check of the WSFC or SQL Server Failover Policy configuration indicates a failed state.By default, a service restart is attempted before a failover to another node is initiated.A timeout in the restart attempt indicates a resource failure.
2) **A failover is indicated.**A Failover Policy check indicates the need for a node failover.
3) **TheSQL Server service is stopped.**If currently running, an orderly shutdown of the SQL Server service is attempted.
4) **The WSFC cluster resource is transferred.**Ownership of the SQL Server cluster resource group and its dependent network and shared storage resources are transferred to the next preferred node owner of the FCI.

5) **SQL Server is started on the new node.**The SQL Server instance goes through its normal startup procedures.If it does not come back online within a pending timeout period, the cluster service puts the resource on this new node in a failed state.

6) **User databases are recovered on the new node.**Each user database is placed in recovery mode while transaction log redo operations are applied and uncommitted transactions are rolled back.

FCI Improvements

Previous versions of SQL Server have offered a FCI installation option; however, several feature enhancements in SQL Server 2012 improve availability robustness and serviceability:

- **Multi-subnet clustering.**SQL Server 2012 supports WSFC cluster nodes that reside in more than one subnet.A given SQL Server instancethat resides on a WSFC cluster node can start if any network interface is available; this is known as an 'OR' cluster resource dependency.

  Prior versions of SQL Server required that all network interfaces be functional for the SQL Server service to start or failover, and that they all existon the same subnet or VLAN.

  Note:Storage-level replication between cluster nodes is not implicitly enabled with multi-subnet clustering.Your multi-subnet FCIsolution must leverage a third-party SAN-based solution to replicate data and coordinate storage failover between cluster nodes.

  For more information, seeSQL Server 2012 AlwaysOn: Multisite Failover Cluster Instance(http://sqlcat.com/sqlcat/b/whitepapers/archive/2011/12/22/sql-server-2012-alwayson_3a00_-multisite-failover-cluster-instance.aspx).

- **Robust failure detection.**The WSFC cluster service maintains a dedicated administrative connection to each SQL Server 2012 FCI on the node.On this connection, a periodical call to a special system stored procedure, **sp_server_diagnostics**, returns a rich array of system health diagnostic information.

  Prior to SQL Server 2012, the primary health detection mechanism for a FCI was implemented as a simple one-way polling process.In this process, the WSFC cluster service periodically created a new SQL client connection to the instance, queried the server name, and then disconnected.A failure to connect, or a query timeout, for whatever reason, triggered a failover with very little available diagnostic information.

  For more information, seesql_server_diagnostics (http://msdn.microsoft.com/en-us/library/ff878233(SQL.110).aspx).

There is now broader support for FCI storage scenarios:

- **Better mount point support.**SQL Server setup now recognizes cluster disk mount point settings.The specified cluster disks and all disks mounted to it areautomatically added to the SQL Server resource dependency during setup.

- **tempdb on local storage.**FCIs now support placement of **tempdb**on local non-shared storage, such as a local solid-state-drive, potentially offloading a significant amount of I/O from a shared SAN.

Prior to SQL Server 2012, FCIs required **tempdb**to be located on a symmetrical shared storage volume that failed over with other system databases.

Note:The location of **tempdb**is stored inthe **master** database, which moves between nodes during failover.It must be on a valid symmetrical file path (drive, folders, and permissions) on all potential node owners, or else the SQL Server service will not start on some nodes.

Database Availability

The high availability capabilities offered by the infrastructure and SQL Server instance-level components work together to implicitly protect hosted databases.An AlwaysOn solution offers an additional set of options for explicitly protecting database data and data tier applications.

AlwaysOn Availability Groups

An *availability group* is a set of user databases that fail over together from one SQL Server instance to another within the same WSFC cluster.Client applications can connect to the availability group's databases through a WSFC virtual network name,known as an *availability group listener*, which abstracts the underlying SQL Server instances.

AlwaysOn Availability Groups rely upon Windows Server Failover Clustering for health monitoring, failover coordination, and server connectivity.You must enable AlwaysOn support on a SQL Server instance that resides on a WSFC cluster node.However, that instance does not have to be a FCI, and it does not require the use of symmetrical shared storage.

For more information, seeOverview of AlwaysOn Availability Groups (http://msdn.microsoft.com/en-us/library/ff877884(SQL.110).aspx).

Availability Replicas and Roles

Each SQL Server instance in the availability group hosts an *availability replica* that contains a copy of the user databases in the availability group.A SQL Server instance can host only one availability replica from a given availability group, but multiple availability groups may reside on the same instance. The SQL Server instance must have dedicated (non-shared) storage volumes.

One of the availability replicas serves in the role of *primary replica*.It is designated as the master copy of the availability group databases and is enabled for read/write operations.

An availability group can contain from one to four additional read-only availability replicas that each separately serve in the role ofa *secondary replica*.

Availability Replica Synchronization

The contents of each database in an availability group are synchronized from the primary replica to each of secondary replicas through a mechanism of SQL Server log-based data movement.For this reason, all databases in the availability group must be set to the full recovery model.

Secondary replicas are initialized with a full backup and restore of the primary replica's databases and transaction logs.As new transactions are committed on the primary replica, the corresponding portion of the transaction log is cached, queued, and then sent over the network to a database mirroring endpoint on each of the secondary replica nodes.

In this manner, new entries in the primary replica transaction log are appended onto each of the secondary replica's transaction logs.Each secondary replica periodically communicates a log sequence number (LSN) back to the primary replica to indicate a watermark of how much of their transaction log has been hardened and flushed to the remote disk.

Note:Each availability replica has its own set of independent transaction *log redo threads* that are not part of the availability replica synchronization process.You may perceive delays in the log redo process on the secondary replicas as data latency.

In addition to having a role of primary or secondary, each availability replica also has an *availability mode*, which governs the coordination of hardening the transaction logsduring a COMMIT TRAN statement:

- **Synchronous-commit mode.** The primary replica commits a given transaction only after all synchronous-commit secondary replicas acknowledge that they have finished hardening their respective transaction logs past that transaction's LSN. An availability group can have up to 2 synchronous-commit secondary replicas.

  Synchronous-commit mode introduces transaction latency on the primary replica databases, but it ensures that there is no data loss on the secondary replicas for committed transactions.

- **Asynchronous-commit mode.**The primary replica commits transactions after hardening the local transaction log, but it does not wait for acknowledgement that an asynchronous-commit secondary replica has hardened its transaction log.An availability group can have up to 4 asynchronous-commit secondary replicas, but no more than a total of 4 secondary replicas of any type.

  Asynchronous-commit mode minimizes transaction latency on the primaryreplica databases but allows the secondary replica transaction logsto lag behind, making some data loss possible.

For more information, seeAvailability Modes (http://msdn.microsoft.com/en-us/library/ff877931(SQL.110).aspx).

The overall health of the data flow between the availability replicas is indicated by the *synchronization state* of each replica.You will most likely experience data loss if you fail over to a secondary replica with a synchronization state of anything other than 'Synchronized' or 'Synchronizing'.

Each secondary replica's synchronization stream has a *session timeout* property.When a secondary replica configured for a synchronous-commit availability mode fails with a session timeout, it is temporarily marked internally as asynchronous.This is done so that the secondary replica failure does not impact hardening of the transaction log on the primary replica.After that secondary replica is healthy and caught back up with primary replica, it automatically reverts to normal synchronous-commit mode operations.

Availability Group Failover

The availability group and a corresponding virtual network name are registered as resources in the WSFC cluster.An availability group fails over at the level of an availability replica, based upon the health and failover policy of the primary replica.

An availability group failover policy uses the **FailureConditionLevel** property to indicate the severity tolerance level for a failure affecting the availability group, in conjunction with the **sp_server_diagnostics** system stored procedure.This same mechanism is used for FCI failover policies.

In the event of a failover, instead of transferring ownership of shared physical resources to another node, WSFC is leveraged to reconfigure a secondary replica on another SQL Server instance to take over the role of primary replica.The availability group's virtual network name resource is then transferred to that instance.All client connections to the involved availability replicas are reset.

Based upon the current health, synchronization state, and availability mode of the replicas, each replica has a composite*failover readiness* state that indicates the potential for data loss.This replica health information is viewable in the AlwaysOn Dashboard, or in the *sys.dm_hadr_availability_replica_states s*ystem view.

Each availability replica also has a configured*failover mode*, which governs replica behavior when failover is indicated.

- **Automatic failover (without data loss)**.This allows for the fastest failover time of any AlwaysOn configuration because the secondary replica transaction log is already hardened and synchronized.Open transactions on the primary replica are rolled back, and the primary replica role is transferred to a secondary replica without any user intervention.

  The primary and secondary replicas must be set to automatic failover mode, and both must be set to synchronous-commit availability mode.The synchronization state between the replicas must be 'Synchronized'.Additionally, the WSFC cluster must have a healthy quorum.

  Automatic failover is not supported if the primary or secondary replica resides on an FCI.This is blocked to prevent a potential race condition between availability group and FCI failovers.

- **Manual failover**.This allows the administrator to assess the state of the primary replica, and make a decision to deliberately fail over to a secondary replica or not.

  Depending upon the availability mode and synchronization state, you have these choices:

  - **Planned manual failover (without data loss).**You can perform this type of failover only if both the primary and secondary replicas are healthy and in a 'Synchronized' state. This is functionally equivalent to an automatic failover.
  - **Forced manual failover (allowing potential data loss).**This is the only form of failover that is possible if the target secondary replica is in asynchronous-commit availability mode, or if it is not synchronized with the primary replica.

    Warning: You should use this failover option in a disaster recovery situationonly.If the primary replica is healthy and available, you should change the availability mode of the involved replicas to synchronous-commit and then perform a planned manual failover.

    For more information, seePerform a Forced Manual Failover of an Availability Group(http://msdn.microsoft.com/en-us/library/ff877957(SQL.110).aspx).

You must perform a manual failover if any of the following conditions are true about either the primary replica or the secondary replica that you want to fail over to:

- Failover mode is set to manual.
- Availability mode is set to asynchronous-commit.
- Replica resides on an FCI.

For more information, seeFailover Modes (AlwaysOn Availability Groups)(http://msdn.microsoft.com/en-us/library/hh213151(SQL.110).aspx).

Note: After a failover, if the new primary replica is not set to the synchronous-commit mode, the secondary replicas will indicate a 'Suspended' synchronization state. No data will flow to the secondary replicas until the primary replica is set to synchronous-commit mode.

Availability Group Listener

An *availability group listener* is a WSFC virtual network name (VNN) that clients can use to access a database in the availability group.The VNN cluster resource is owned by the SQL Server instance on which the primary replica resides.

The virtual network name is registered with DNS only during availability group listener creation or during configuration changes.All virtual IP addresses that are defined in the availability group listener are registered with DNS under the same virtual network name.

To use the availability group listener, a client connection request must specify the virtual network name as the server, and a database name that is in the availability group.By default, this should result in a connection to the SQL Server instance that is hosting the primary replica.

At runtime, the client uses its local DNS resolver to get a list of IP addresses and TCP ports that map to the virtual network name.The client then attempts to connect to each of the IP addresses, until it is successful, or until it reaches the connection timeout. The client will attempt to make these connections in parallel if the **MultiSubnetFailover** parameter is set to true, enabling much faster client failovers.

In the event of a failover, client connectionsarereset on the server, ownership of the availability group listener moves with the primary replica role to a new SQL Server instance, and the VNN endpoint is bound to the new instance's virtual IP addresses and TCP ports.

For more information, seeClient Connectivity and Application Failover(http://msdn.microsoft.com/en-us/library/hh213417(SQL.110).aspx).

Application IntentFiltering

While connecting through the availability group listener, the application can specify whether its intent is to both read and write data or whether it will exclusively perform read-onlyoperations. If not specified, the default application intent for the client is read-write.

For the primary role and secondary role of each availability replica, you can also specify a *connection access* property that will be used as a connection-level filter on the client's application intent.By default, invalid application intent and connection access combinations result in a refused connection.SQL Server should filter out client connection requests using the following rules.

While the availability replica is in the primary role, and connection access is equal to:

- **Allow any application intent.**Do not filter any client connections for application intent.
- **Allow only explicit read/write intent.**If client specifies read-only, reject connection.

While the availability replica is in the secondary role, and connection access is equal to:

- **No connections allowed.**Refuse all connections; replica is used only for disaster recovery.
- **Allow any application intent.**Do not filter any client connections for application intent.
- **Read-only application intent.**If client does not specify read-only, reject connection.

For more information, seeConfigure Connection Access on an Availability Replica(http://msdn.microsoft.com/en-us/library/hh213002(SQL.110).aspx).

Application Intent Read-Only Routing

A key value proposition for AlwaysOn Availability Groups is the ability to leverage your standby hardware infrastructure for purposes other than disaster recovery.By configuring one or more of your secondary replicas for read-only access, you can offload significant workloads from your primary replicas.

Workloads that can be readily adapted to run off of a read-only secondary replica include:reporting, database backups, database consistency checks, index fragmentation analysis, data pipeline extraction, operational support, and ad-hoc queries.

For each availability replica, you can optionally configure a sequential *read-only routing list* of SQL Server instance endpoints to be applied while that replica is in the primary role.If present, this list is used to redirect client connection requests that specify read-only application intent to the first available secondary replica in the list that satisfies the application intent filters noted earlier.

Note: The read-only routing redirection is performed by the availability group listener, which is bound to the primary replica. If the primary replica is offline, client redirection will not function.

For more information, seeConfigure Read-Only Routing on an Availability Group (SQL Server)(http://msdn.microsoft.com/en-us/library/hh653924(SQL.110).aspx)

Availability Improvements – Databases

SQL Server 2012 has a number of feature enhancements that are specific to database configuration and capabilities.

The followingimprovementreduces recovery time:

- **Predictable Recovery Time.**You can set a *target recovery time* interval per database, which is used to control the scheduling of a background CHECKPOINT command.This *indirect checkpoint* occurs periodically, based upon estimated time needed to recover the transaction log in the event of a restart or failover.This has the effect of smoothing I/O out to roughly equal proportions for each checkpoint, and increasing recovery time (RTO) predictability.

  Prior to SQL Server 2012, background CHECKPOINT commands were issued on a fixed interval, irrespective of transaction volume or load, which could lead to unpredictable recovery times.

  For more information, seeDatabase Checkpoints (http://msdn.microsoft.com/en-us/library/ms189573(SQL.110).aspx).

These improvements mitigate common scenarios that can drive planned downtime:

- **Online index operations for LOB columns.**Indexes that contain columns with **varbinary**(max), **varchar**(max), **nvarchar**(max), or XML data types can now be rebuilt or reorganized online.
- **Online schema modification for new NOT NULL columns.**If a new NOT NULL column is added with a default value to a SQL Server 2012 database table, only a schema lock is required to update system metadata; all rows do not have to be populated during the ALTER TABLE statement.

  SQL Server will physically persistthe default column value only if a row is actually modified or re-indexed.Queries return the default value from metadata, unless an actual column value exists.

There is an example of broader support for storage scenarios:

- **Automatic Page Repair**.Certain types of storage subsystem errors can corrupt a data page, making it unreadable.AlwaysOn Availability Groups can detect and automatically recover from these types of errors by asynchronously requesting and applying a fresh copy of the affected data pages from a different availability replica.

  Similar functionality existed prior to SQL Server 2012 for database mirroring, but it is now enhanced to support multiple replicas.

  For more information, seeAutomatic Page Repair(http://msdn.microsoft.com/en-us/library/bb677167(SQL.110).aspx).

Client Connectivity Recommendations

Follow these guidelinesto enable client applications to take full advantage of Microsoft SQL Server 2012 AlwaysOn technologies:

- **AlwaysOn-aware client library.**Use a client library that supports thetabular data stream (TDS) protocol version7.4 or newer.This should provide the desired client-side functionality for AlwaysOn features.Example client libraries include the Data Provider for SQL Server in .NET Framework 4.02, and the SQL Native Client 11.0.

- **Connection provider property:MultiSubnetFailover = True.**Use this keyword in your connection strings to enable client libraries to attempt to connect in parallel to all IP addresses that are registered for the availability group listener or the FCI that has IP address in multiple subnets.

- **Connection provider property:ApplicationIntent = ReadOnly.**Where practical, offload read-only workloads from your primary replica onto the secondary replicas.

- **Legacy client connection timeout.**Legacy client database libraries do not implement parallel connection attempts, so when multiple IP addresses are present, they try to connect to each of them sequentially, until they encounter a TCP timeout, or until they make a successful connection.

  You should adjust your connection timeout on legacy clients to accommodate the potential sequential timeouts and retries when multiple IP addresses are present, to a value that is at least 15 seconds + 21 seconds for every secondary replica.

Conclusion

This white paper has established the baseline context for how to reduce planned and unplanned downtime, maximize application availability, and provide data protection using SQL Server 2012 AlwaysOnhigh availability and disaster recovery solutions.

Many of the business drivers and challenges of planning, managing, and measuring a highly available database environment can be quantified and expressed as Recovery Point Objects (RPO) and Recovery Time Objectives (RTO).

SQL Server 2012 AlwaysOn providescapabilities at the infrastructure, data platform, and database level that can help your organization address common high availability and disaster recovery scenarios, in a manner that can be well-justified using RPO and RTO goals.

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.

◊Version 1.1, 21 February 2012.