the Availability Digest

www.availabilitydigest.com

Choosing a Business Continuity Solution

Part 3 – High Availability Architectures

September 2011

In the previous two parts of this series,[1] we defined a variety of availability metrics and pointed out that data replication was fundamental to providing the data redundancy needed to achieve high and continuous availability. We explored various data replication technologies and described their strengths and weaknesses.

Of particular interest are unidirectional and bidirectional asynchronous and synchronous replication engines. In Part 3, we look at a variety of highly available system architectures that use these data replication technologies to achieve a wide range of availability characteristics.

Data Replication Provides High and Continuous Availability

We distinguish here between high availability and continuous availability. A system that provides high availability can recover from an outage quickly enough so that no serious damage is done to the enterprise. The required recovery time is a function of the criticality of an application. Recovery time measured in minutes is typically sufficient to provide high availability to critical applications.

A system provides continuous availability if it recovers so quickly that no one is aware that there has been an outage or at least is not inconvenienced by the outage. Recovery time in seconds generally qualifies for continuous availability.

Data replication not only satisfies the five fundamental requirements of availability, described earlier and summarized below, but it also can satisfy very small RTOs and very small or zero RPOs.

Redundancy

Since the intent of data replication is to copy data from a source database to a target database, it inherently supports redundancy. There will be at least two synchronized copies of the database in the application network.

Furthermore, since the replication engine can be configured to maintain the target database as a complete and consistent copy of the source database, the target database can be used for application processing as well. Thus, replication not only supports data redundancy, but it also supports processing redundancy.

[1] This series of articles is a reprint of a Gravic, Inc., white paper and is published with the permission of Gravic. See the Gravic web site for their other white papers.

Isolation

A redundant system based on data replication ensures isolation of the redundant components. Since the two (or more) data-processing sites interact only via data replication, a problem at one site will not affect the other sites.

Furthermore, since the sites are loosely coupled and stand on their own, they do not have to be identical. They can use different hardware, different application and operating-system versions, and even different database schemas.[2] It is only necessary that each site be a fully operational site in its own right.

Dispersion

Data replication inherently supports geographic dispersion. The sites can be hundreds or thousands of miles apart. However, the farther apart the systems are, the greater will be the replication latency. Therefore, it is important to select replication architectures that minimize any latency that they add to replication processing.

Failover

One site can automatically detect a fault in another site by monitoring the data flow in the replication channel. For instance, during idle times, the source system can send heartbeat, status, or “I’m alive” messages to the target system. If the target system detects a lack of replication or heartbeat traffic, it can initiate the failover process.

If the target system is also actively processing transactions, or if it is kept in a ready state to do so, failover is simply a matter of switching users from the failed site to the operational site. It is known that the surviving site is operational. There is no decision time involved – just do it.

Testing

Compared to active/passive systems, testing failover is much simplified with active/active systems. Testing an active/passive architecture is difficult and risky and requires an application outage as failover is tested. Bringing up the backup system requires loading (if magnetic tape backup or virtual tape backup is used) or activating the backup database, starting the applications, switching the networks, and testing the system. It is expensive since critical personnel must be on-site or on-call in case something goes wrong.

Active/passive failover testing can take hours, during which the application is down. And it is risky. What if the failover doesn’t work? What if the maintenance window is exceeded? What if the primary system doesn’t come back up? As a result, full failover testing is often not done. The driving factor when a failure does occur and the backup must be brought online is often faith. No wonder management is often reluctant to make the call to fail over.

In an active/active system, the backup is always operational. Testing it only requires periodically sending test or verification messages to the backup system. Alternatively, a subset of users can be periodically switched over to the known good backup system, a process that typically can be done in seconds or subseconds. Thus, failover testing can be performed often and with little risk. Of course, if all nodes are actively processing transactions, it is always known that all nodes are performing properly. Every transaction is, in effect, a test message.

[2] If this capability is important in your application, be sure that your data-replication vendor supports heterogeneous replication between the systems and databases that you have.

RTO

As can be seen by the failover description above, data replication allows recovery from a system fault to be accomplished in seconds or less. Recovery is simply a matter of rerouting user or transaction requests to a surviving node in the application network.[3] Therefore, RTOs measured in subseconds or seconds can be achieved.

RPO

If asynchronous replication is used, RPOs in the order of tens or hundreds of milliseconds can be achieved. If synchronous replication is used, no data is lost following a source-node failure. An RPO of zero is achieved.

However, it should be noted that whereas synchronous replication will lose no data, there is a window of uncertainty following a source-node failure. Since a synchronous replication engine typically safe-stores or tentatively applies the transaction changes on the target system before voting to commit, there is a brief window during which a source-node failure will leave the target transaction in doubt as to whether it was committed or aborted on the source system. Upon takeover, the target system will not know what to do with the transaction. This is a hung transaction and must be resolved manually or by business rules.[4]

The Business Continuity Continuum

Business Continuity – a Summary

The various business continuity approaches that we have described earlier are summarized in Figure 1. This diagram plots RPO versus RTO and shows where each methodology fits. The horizontal axis shows improving RTO as it trends toward zero (or unnoticeable) recovery time, and the vertical axis shows improving RPO as it trends toward zero data loss following a node failure. System continuity increases as you move up and to the right.

Hence, most availability implementations are in the upper right quadrant of the figure using one of the data-replication architectures. Best practices are pushing implementations from active/passive to active/active and (assuming the appropriate technology exists) from asynchronous replication to synchronous replication.[5]

Though the figure reflects the full range of business continuity strategies, we are only concerned in this white paper with the data-replication architectures that can provide RTOs measured in minutes to subseconds and RPOs measured in seconds to zero. They reflect the trends that are occurring in the marketplace as companies realize that they need even those services once deemed ancillary, such as email and Internet access, to be always available.

A broad range of techniques exists for minimizing data loss and unplanned downtime, from tape backups to active/active systems. We briefly review them next to place them in perspective.

[3] Users can be switched using user redirection, network redirection, or server redirection. See Achieving Fast Failover in Active/Active Systems, Parts 1 and 2, *Availability Digest*; August and September, 2009.

[4] As a practical matter, if the target system knows that all I/O events were successfully applied to the source database, it can assume that the source system issued the commit and do the same. Coordinated-commit replication engines know this information.

[5] Verify the range of support with your replication-engine vendor.

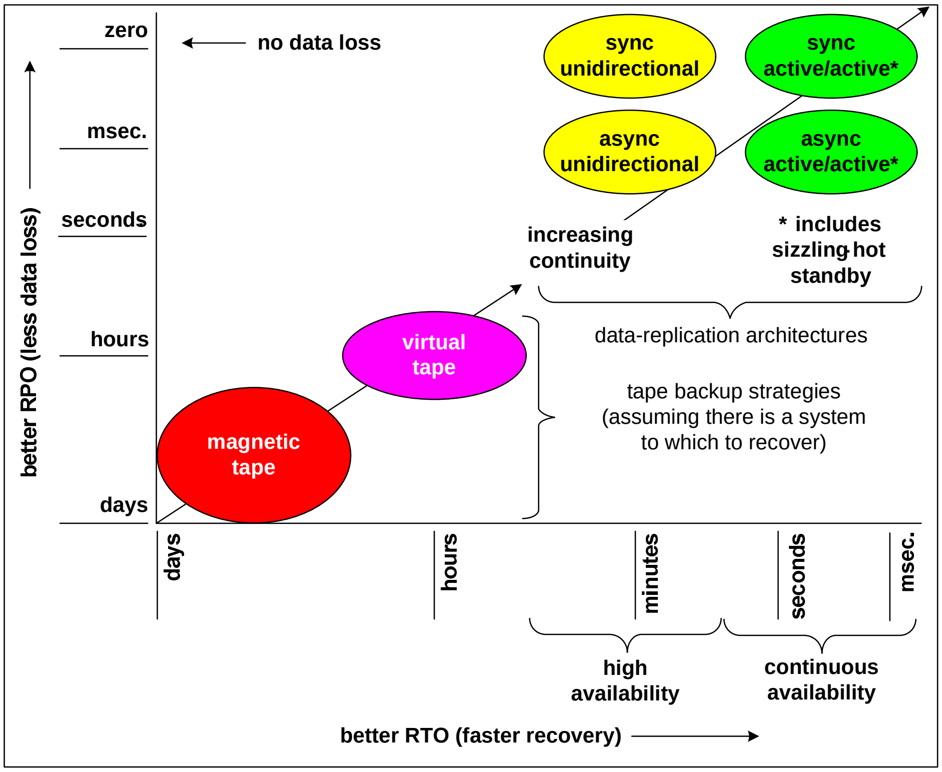

Figure 1: The Business Continuity Continuum

Disaster Recovery

Disaster recovery is the ability to *recover* from a disaster, even if it takes hours or days.

Tape Backup

From the earliest days of commercial computing, systems have used magnetic tape as a backup medium (Figure 2). Operations staff take full backups followed by intermediate incremental backups taken periodically, such as every day. At the time of failure, the recovery point is the last backup. Therefore, the time between backups leads to very long RPOs – hours to days. Furthermore, the time involved in retrieving the tapes, recovering the database, bringing up applications, and testing the resulting backup system also results in RTOs of hours to days, assuming that you have a system to which to recover.

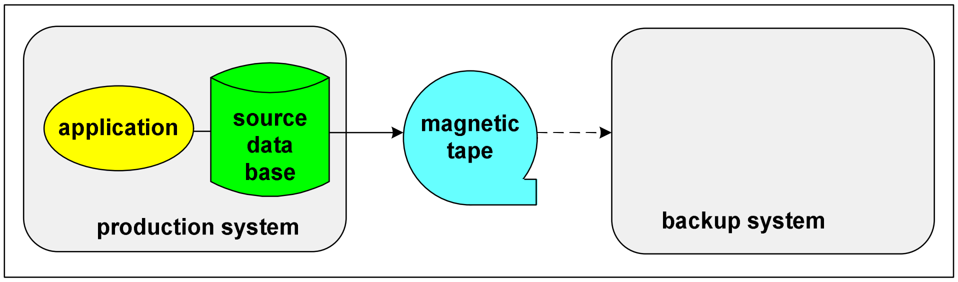

Figure 2: Magnetic Tape Backup

Virtual-Tape Backup

Virtual tape replaces magnetic tape with disk images of the database communicated to an offsite disk-storage system (Figure 3). Because the need to handle and store large numbers of tapes is eliminated, the frequency of backups can be increased, significantly reducing RPOs to hours. When a recovery needs to be performed, the loading of the backup database is much faster, though applications must still be started and the system tested, leading to RTOs typically measured in hours or more.

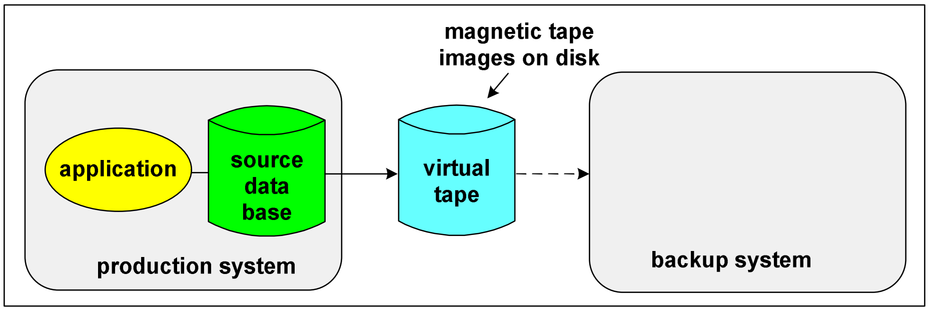

Figure 3: Virtual Tape Backup

Because of the lengthy requirement to procure an alternate system, load it, synchronize its database, and bring the application environment online following the loss of the primary system, magnetic-tape and virtual-tape approaches do not meet high- or continuous-availability requirements. We will not consider these technologies further in this paper.

Clusters and Virtualization

Clusters[6] and virtualization[7] can lead to very good availabilities within the data center or campus environment (five 9s is often quoted), but these techniques apply only to collocated systems. The backup system and the storage shared between the systems typically must be in the same data center or otherwise located in close proximity with extensive high-speed interconnects (Figure 4). Therefore, these technologies do not satisfy the dispersion attribute of highly available systems. They cannot survive a local disaster.

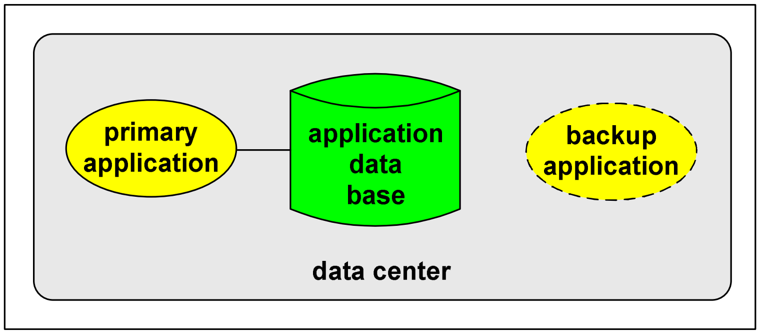

Figure 4: Clusters and Virtualized Environments

Unidirectional Replication

Unidirectional data replication is the simplest form of data replication (Figure 5). An active node processes all transactions and replicates the database changes that it makes to a remote standby

[6] Active/Active Versus Clusters, *Availability Digest*; May 2007.
http://www.availabilitydigest.com/public_articles/0205/clusters.pdf

[7] Fault-Tolerance for Virtual Environments – Parts 1 to 3, *Availability Digest*; March 2008.
http://www.availabilitydigest.com/public_articles/0303/virtualizatiion_1.pdf

database. Therefore, the two databases are in (or are nearly in) synchronization. With asynchronous replication, only the data in the replication pipeline at the time of a source node failure will be lost, thus supporting subsecond RPOs. With synchronous replication, no data will be lost following an active node failure (RPO of zero).

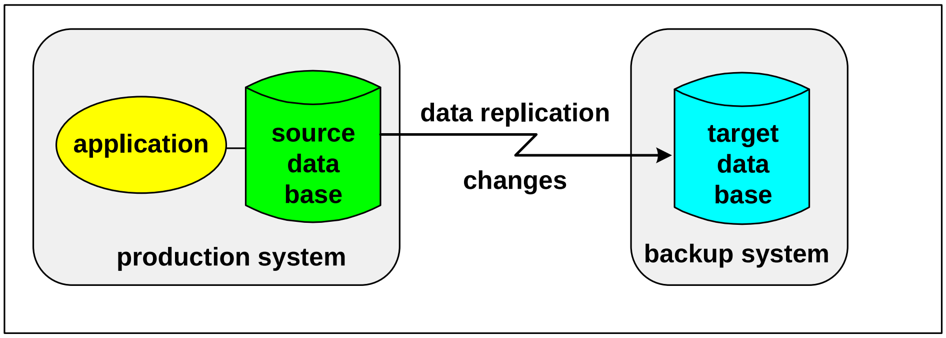

Figure 5: Unidirectional Replication

Applications may be up and running in read-only mode in the standby node so that the standby database may be actively used for query and reporting purposes (we call this a *hot standby*). Should the active node fail, the applications at the backup node can remount the database for read/write access and can take over the role of the original active node. This typically takes only a few minutes, leading to RTOs measured in minutes.

Thus, unidirectional architectures provide high availability – RTOs measured in minutes and RPOs measured in subseconds (or zero if synchronous replication is used).[8]

Disaster Tolerance

The above systems meet the needs of disaster recovery but not necessarily disaster tolerance. We say this because their recovery times are measured in minutes or more. If users are down for several minutes, or even worse, hours or days, the outage has certainly affected them. To be disaster-tolerant, recovery must be so fast that users are unaware of the outage or at least are not overly inconvenienced by it. Recovery times measured in seconds or less with little or no data loss qualify a system to be called disaster-tolerant.

Disaster tolerance requires that there be a backup node that can take over in subseconds or seconds in the event of an active-node failure. We now describe two data-replication configurations that can satisfy this requirement.

Sizzling-Hot Standby

A sizzling-hot standby is similar to a standby or passive node using unidirectional replication, as described earlier, except that it is immediately ready to start processing transactions (Figure 6). This means that its applications are running and have the local copy of the application database already open for read/write access. Using data replication, the sizzling-hot standby can immediately take over processing because its local database is synchronized with the active database and is completely consistent and accurate.

Should the active node fail, all that is required for failover is to switch the users or their transactions to the standby node. This can be done in subseconds to seconds, leading to very small RTOs.

[8] In some cases, the application on the passive target node can already have the target database open for read/write access, thereby improving recovery time. We call this a *sizzling-hot standby* system and describe it next.

To operate in this mode, it is essential that the replication engine allows the application processes to also open the target database for read/write access.

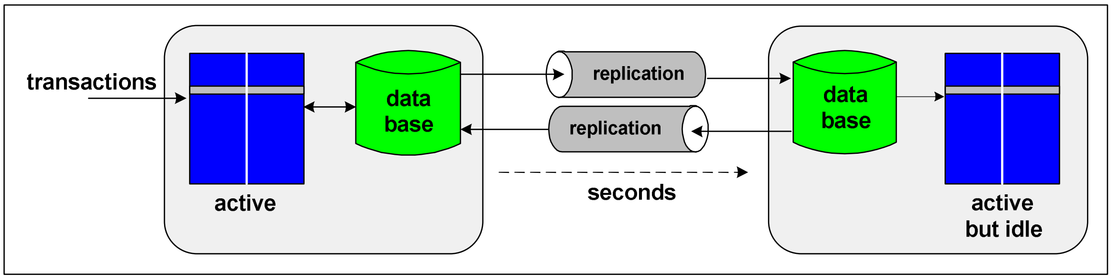

Figure 6: Sizzling-Hot Standby with Optional Reverse Replication

The sizzling-hot standby configuration has another big advantage over the disaster-recovery systems described earlier, and that is the absence of failover faults. In active/passive systems, the standby system is not actively involved in the application. Therefore, one does not know whether it is really operational. Should a failover attempt be made to a nonfunctioning backup system, the application is down. This is known as a *failover fault*.

With a sizzling-hot standby, one knows that the backup node is working (it can be periodically exercised by submitting test or verification transactions to the application to ensure this). Consequently, failover can be automated, which is a requirement if very short RTOs are to be satisfied.

Finally, the sizzling-hot standby can be optionally configured with reverse replication up and running so that it has a backup as soon as the old primary node is recovered. With reverse replication enabled, the standby will queue the changes that it is making to its copy of the database so that the failed node can be resynchronized upon recovery.

This configuration can achieve a zero RPO if synchronous replication is used or RPOs measured in tens or hundreds of milliseconds if asynchronous replication is used. If failover is automatic, RTOs measured in subseconds or seconds can often be satisfied.

Active/Active

An active/active configuration takes the sizzling-hot standby arrangement one step further. All nodes in an active/active network may be simultaneously processing transactions for the same application using their local copies of the application database (Figure 7). Bidirectional data replication is configured between each node pair so that any change that an application makes to its local copy of the database is immediately replicated to the other nodes in the application network. Therefore, all nodes have the same view of the application database.

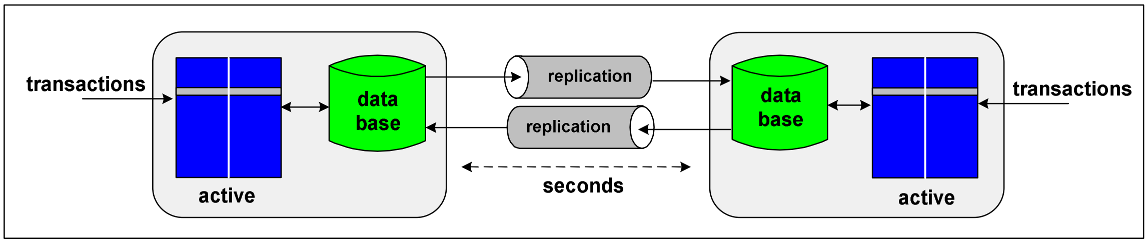

Figure 7: Active/Active

One issue with active/active systems using asynchronous replication is that of data collisions. If two applications on different nodes attempt to change the same row within the replication latency of the data-replication engine, they will both succeed, not knowing that the application on the other node is also doing so. Each will then replicate its change to the other node, overwriting the change to the data item originally made at that node. Now both databases are different, and both are wrong. This is known as a *data collision.* Data replication engines that support active/active architectures provide mechanisms to detect and resolve data collisions.

In an active/active environment, data collisions will not occur if synchronous replication is used because the locking of data items ensures that only one change can be made to all data copies at any one time.

Eliminating Planned Downtime

The above discussions have focused on eliminating *unplanned downtime*. But what about *planned downtime*?

The answer is simple. Using either of the continuously available, disaster-tolerant architectures that we have described – sizzling-hot standby or active/active – a node to be serviced can easily and safely be taken out of service by moving its users to other nodes in seconds. The node can then be upgraded, resynchronized, and returned to service, continuing once again to provide application services to its users. Upgrades can be rolled through the application network in this manner.[9]

A highly available system using unidirectional replication can be upgraded in the same way, except that users may be down for minutes or longer as the backup system is brought into operation.

Summary

We have now applied the various data replication technologies to achieve a variety of high availability architectures. But which one matches your application's needs from both an uptime and a cost perspective? We look at this question in Part 4 of this series.

[9] Using Shadowbase to Eliminate Planned Downtime with Zero Downtime Migrations, Gravic White Paper.