Using HPE Shadowbase Software to Eliminate Planned Downtime via Zero Downtime Migration

A Gravic, Inc. White Paper

Executive Summary

Gone are the days when a company could say that computer system downtime *may be* hazardous to its health. In ever increasing cases today, downtime definitely *is* hazardous to a company's health. *High availability* and even *continuous availability* are required for many applications. We define a highly available system as one that is down no more than a few minutes a year. We define a continuously available system as one that recovers from a fault so quickly that users never notice an outage.

In some cases, applications require such availability because of the extraordinary value of the transactions that must be processed. This situation is common in financial and brokerage applications. In other applications, outages can imperil life or property. Emergency 911 systems are a good example. Beyond cost in terms of dollars and safety, outages can cause loss of reputation and customer loyalty, and bad publicity. Any online retailer knows that an unreachable web site causes potential customers to click elsewhere. In other cases, regulatory compliance drives availability needs. The European Central Bank (ECB), for example, is suggesting highly available no-data-loss architectures as an option, which is typically the first step before regulatory requirements are issued.

Both high availability and continuous availability require the minimization or elimination of *unplanned downtime* due to unexpected failures such as those caused by disasters. They also require the elimination of *planned downtime* such as that needed for an operating system or hardware upgrade or for installation of new application versions. After all, users are down in either case. Whenever changes are made to a system – whether these changes affect hardware, software, networks, or operating procedures – there must be a process to make these upgrades without denying users access to their IT services. When upgrades are undertaken without denying application services to users, it is called Zero Downtime Migration (*ZDM)*, which is the focus of this paper.

In order for a system to achieve high availability or continuous availability, it must meet three requirements:

1. **Redundancy:** Every system component, be it a processing node, a storage system, or a network device, must have a backup that is geographically distant to avoid dual failures due to a common event.
2. **Fast and Reliable Failover:** In the event of a component failure, the failover to the backup component must be reliable. It should also be fast enough that users are unaware or at least are minimally impacted by the fault.
3. **Durable Data:** No critical data is lost as a result of a system fault.

Active/passive systems, where a passive system is available to take over processing if the production system fails, meet the first requirement; however, they fall short with respect to the other two requirements. Failover can take hours and is too often unsuccessful because of its complexity, and difficulty of testing. Depending upon the data replication architecture deployed, all data since the last backup may also be lost.

Active/active system architectures, on the other hand, fulfill the first two requirements. An active/active system[1] comprises two or more geographically dispersed nodes that are actively participating in a common application. That is, each node is actively processing and sharing the application load with the other nodes. The production database is replicated between nodes, in both directions (*bi-directional replication*). If a node fails or needs to be brought down for maintenance purposes, transactions or users are simply switched from the failed or downed node to the surviving nodes, a switch that is accomplished in subseconds or seconds. Users connected to the surviving nodes see no interruption at all. This capability is used to advantage to roll upgrades through the application network without impacting the user's access to key IT services.

The choice of the replication approach adopted determines whether the third requirement is met. Asynchronous replication architectures may lose some amount of data after a fault, whereas synchronous replication architectures eliminate data loss following a fault. The choice of which to implement and deploy

[1]For more information, see the Gravic white paper, *Achieving Century Uptimes with HPE Shadowbase Active Active Technology*.

depends upon a variety of factors, including application design, customer business needs, and the hardware and software selected to host the application. But most importantly this decision depends upon the value of the data. For some business services, some data loss may be acceptable, whereas for others data loss cannot be tolerated at all. This paper delves into these topics in more detail and refers the reader to additional resources for more information.[2]

The HPE Shadowbase suite of products from Gravic, Inc., provides the facilities needed for zero downtime migration for both active/active and active/passive architectures. They include the HPE Shadowbase data replication engine, the SOLV online-load facility, and the SOLV validation and verification utility. Taken together, these products offer the means to eliminate planned downtime for system, application, site, or database upgrades and to verify that the upgrade was successfully and accurately performed, thereby eliminating all the associated business costs and risks of an IT service outage.

[2]For more information, see the Gravic white paper, *Choosing a Business Continuity Solution to Match Your Business Availability Requirements*.

Table of Contents

Table of Figures

Using HPE Shadowbase to Eliminate Planned Downtime via Zero Downtime Migration

Downtime Is Hazardous To Your Health

Gone are the days when a company could say that computer system downtime *may be* hazardous to its health. In today's always-on world, downtime definitely *is* hazardous to a company's health. *High availability* and even *continuous availability* are required for many applications if a company is to avoid the significant costs of service outage. We define a highly available system as one that is down no more than a few minutes a year. We define a continuously available system as one that recovers from a fault so quickly that users never notice an outage, or are at least not inconvenienced by one.

In some cases, outages can be very costly because transactions have extraordinary value. Banks report losses in the several hundred thousands of dollars when their money-transfer systems fail and lose a few seconds of transactions. In addition, banks may be subject to heavy regulatory fines. Large brokerage firms face potential losses in the tens of thousands of dollars per minute if they suffer an outage. A poll of other enterprise users indicates that the average cost of their downtime is between $600,000 and $2,000,000 per hour.[3]

> On Tuesday, June 19, 2012, operations at the Royal Bank of Scotland, NatWest, and Ulster Bank came to a halt. Millions of bank customers were affected; they could not receive their salaries or pension payments, pay their bills, or use the banks' online services. The outage spilled over to customers of other banks when expected payments could not be made. The problem was a software upgrade that had gone terribly wrong, and it took two weeks before operations returned to normal. The outage costs were estimated to be between £50 million and £100 million.[4]

In other applications, outages can endanger life or property, such as outages suffered by emergency 911 systems. When electronic health records (EHRs) – a recent initiative that gives doctors and nurses up-to-date information on their patients – is down, it can be life-threatening as well.

> In August 2014, EHRs for thousands of patients became unavailable due to an outage in the Practice Fusion cloud system, where the records are stored. The doctors could not see patients' lab results or the medications that they were taking. One New York doctor who uses Practice Fusion's software said the outage could have been "a disaster if you're dealing with life and death situations." The outage was caused by an upgrade to address network access problems.[5]

Beyond cost in terms of dollars and safety, outages can cause loss of reputation and customer loyalty, and bad publicity – the so-called "CNN moment."[6]

> Verizon shocked the cloud community when it announced on January 5, 2015, that it would take down its Cloud Compute and Cloud Storage systems for routine upgrades for up to 48 hours starting in the early morning hours of Saturday, January 10. There was no plan to move active virtual machines to other operating physical servers in other data centers during the maintenance period. All hosted virtual machines (VMs) in all data centers would be unavailable. The Twitter universe was filled with angry and sarcastic comments about Verizon's lack of concern for the availability of its cloud services.[7]

These three examples present very critical outages following an upgrade. Was the testing of these upgrades properly planned and executed? Was enough time allowed for a thorough shakedown of a new system before putting it into service? What if the upgrade failed after it was put into service? Was a fallback plan established and tested? Often the necessary testing is not adequately completed because the business service is unavailable during the upgrade, and in the rush to restore the service, corners are cut. Whatever the reasons, an upgrade does not have to result in headlines, if properly planned for and executed.

[3] Network Computing, The Meta Group, Contingency Planning Research
[4] *The Availability Digest*, July 2012
[5] *The Huffington Post*, August 19, 2014
[6] Every CIO's worst nightmare is to have an outage that becomes a media lead story.
[7] *The Availability Digest*, February 2015

High Availability and Continuous Availability

High availability and continuous availability require not only the minimization or elimination of *unplanned downtime* due to unexpected failures but also the minimization or elimination of *planned downtime*. After all, users are down in either case. Whenever changes are made to a system – whether the changes affect hardware, operating system or application software, networks, database versions, or operating procedures – there must be a process in place to enable upgrades without denying users access to their application services. We call this process Zero Downtime Migration, or ZDM, which is the focus of this paper.

Even if some planned downtime is permissible – perhaps there is an evening or weekend maintenance window for accomplishing the upgrade – the commissioning of the newly upgraded system must avoid the risky *big-bang* syndrome of sudden deployment. Will the upgraded system really work in the production environment? If it doesn't, is there a fallback plan? Can fallback be accomplished within the maintenance window? Will the fallback plan work? Will any data be lost when a fallback occurs?

To avoid the big-bang syndrome, the upgraded system must be thoroughly tested prior to putting it into production so that it is a *known-working* system. Users should then be moved slowly to the new system over a period of time. As more and more users are moved, and as the new system carries more and more of the application's load, confidence in the system is gained through real-world experience, not through an aseptic or incomplete testing environment.

Even if your system and application environment cannot avoid a big-bang moment, we discuss methods that at least lessen the risk and provide fallback approaches that work without losing any application data. When a system must be up 24x7, how do we perform an upgrade without denying service to users? How do we avoid or at least mitigate the big-bang syndrome? We address these critical questions in this paper and show you how the HPE Shadowbase product suite from Gravic, Inc. provides you with the tools to truly eliminate planned and unplanned downtime to achieve either high or continuous availability.

What Do We Mean By High Availability?

There is much talk in the industry today about *high availability*. High availability generally means that a system is down for an average of only a few minutes a year, often accomplished via cluster technology or via other approaches that yield failover times to a backup system measured in minutes.

What Do We Mean By Continuous Availability?

High availability is not good enough for many applications. Applications in which downtime costs hundreds of thousands of dollars per minute, applications whose outages bring a business to its knees, or applications that impact the safety of life or property cannot afford to be down minutes per year. They must always be available. For instance, if a stock exchange loses its trading system, the trading day is then brought to a halt. The unavailability of a 911 system prevents help from being sent despite the fact that the system recovered in a few minutes. These systems require *continuous availability*.

Of course, outages can never be totally prevented; something is always bound to fail. However, if the recovery from the outage is so fast that no one notices the outage, and if no data (or only an acceptably small amount of data) is lost as a result of the outage, in effect, continuous availability is achieved. In other words, *let it fail but fix it fast.*

How is High Availability or Continuous Availability Achieved?

In order for a system to achieve high availability or continuous availability, it must have three important characteristics:

Redundancy

Every component in the system must be backed up by another equivalent or reasonably similar component that takes over if the primary component fails. Redundant components should be geographically separated so that no common disaster takes down both of them simultaneously. Redundancy applies to processing systems, storage systems, networks, and any other facility critical to the operation of the application.

Fast and Reliable Failover

If a component fails, the failover to the backup component must be fast enough to meet the system availability requirement.[8] Failover times measured in minutes generally qualify a system as being highly available. Failover times measured in sub-seconds or seconds generally comply with the notion of continuous availability. Failover must also be reliable. A *failover fault*, in which the backup component fails to properly take over, violates high and continuous availability.

Durable Data

The availability of a processing infrastructure is of no value if it does not have access to the data that it needs to fulfill its function. No critical data must be lost due to a component fault and the subsequent failover to the component's backup. In many cases, regulatory authorities are moving in this direction. For instance, the European Central Bank has alluded to this requirement. The "no data loss" dictum is often voluntary today but may become mandatory in the near future. Even if not mandatory from a regulatory point of view, zero or little data loss may still be an absolute requirement from a business point of view.

Active/Backup Systems

The legacy technology for ensuring business continuity, still in use today, employs the use of active/backup systems. In an active/backup configuration, a second backup system stands ready to take over operations if the active system fails, as shown in Figure 1. With respect to production processing, the backup system is idle. It is not actively engaged in production processing, though it may be hosting other applications.

If the backup system needs to take over the processing functions, it must have a reasonably current copy of the application database to use. Classically, magnetic tape was used to take backups (typically, daily full and incremental backups) of the application database so that the database could be restored on the standby system if necessary. If the active system failed, all of the data generated since the last backup was lost. Thus, hours or days of data could be lost upon the failure of the primary system; and it could take hours or days to bring up the standby system.

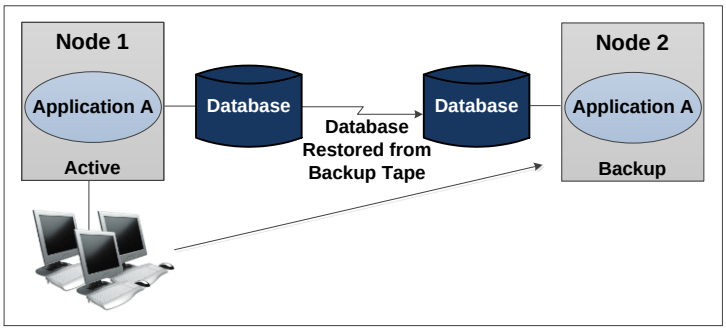

Figure 1 – An Active/Backup System

This situation was significantly improved with the advent of virtual tape, in which backups were made to a remote disk rather than to magnetic tape. With this technique, backups could be taken more frequently. The amount of lost data and the time to recover was reduced from days to hours. This process is much better, but still too long for most critical applications.

Another major problem with active/backup systems that are backed up with tape technology is that maintenance and upgrade activities cause significant downtime. In order to upgrade the active production system, it must be taken out of service. This downtime means pausing data processing while the active database is loaded onto the backup system and then bringing the backup system into service. This process typically requires hours to perform, during which data processing services are not available to the users. This long process must also be performed in the event of an unplanned outage.

Active/Passive Systems

Today's technology uses data replication to keep the standby system's database synchronized in real-time with the source database. Because the standby system now has a fully loaded and synchronized database, it is typically called a *passive* standby system rather than a *backup* system. It is passively standing by, ready to take over processing if the active system fails, as shown in Figure 1. Data replication has improved data protection and recovery times significantly for active/passive systems compared to the older, tape-based active/backup systems, reducing hours or days to minutes.

[8]The maximum allowable failover time that can be tolerated is called the *Recovery Time Objective*, or *RTO*.

Data replication keeps the database copies synchronized.[9] That is, when one node makes a change to its copy of the application database, that change is immediately replicated to all of the other database copies so that all are in the same state.

Data replication may be synchronous or asynchronous. *Asynchronous* replication, in which changes are replicated after the fact from a change queue, may lose some small amount of data (typically measured in the tens or hundreds of milliseconds) if a node fails since data in the replication pipeline may be lost. However, *synchronous* replication – which ensures that all changes are applied across the application network (both to the active database copy and to the passive database copies) or that none are – does not lose any data following a node failure, because if all nodes cannot apply the data, then none apply.

Also, since the passive system always has a current database mounted, it is prepared to take over processing quickly if the primary system fails. If the applications are already up and running on the passive system, transactions are rerouted or users are reconnected to the passive system. Recovery is accomplished in minutes if not seconds rather than in hours. Consequently, data replication in active/passive configurations provides the three requirements for *high availability* – redundancy, fast failover, and durable data.[10]

Using data replication, the problem of planned downtime is eliminated or significantly mitigated when using the ZDM procedures described later. To perform maintenance, the passive system is taken offline and upgraded. After thorough testing, it is put back into service and takes over the role of the primary system. The primary system is now upgraded and thoroughly tested and returned to service as the primary. Each switchover only causes a small amount of user downtime at most since switchover is always to a known working system.

The advantages of this upgrade technique can be used even if the system being upgraded is a single system. For instance, a second system is rented and synchronized with the primary system. The primary system is then taken down, upgraded, synchronized with the online rented system, and returned to service. At this point, the rented system is returned. If a single system is to be replaced with a new system, a similar technique is used. Bring the new system up with all upgrades, synchronize it with the old system, and then put the new system into service.

This paper describes techniques for using data replication to eliminate the planned downtime component of system outages when employing an active/passive architecture or an active/active architecture, as described next.

Active/Active Systems

Active/active system architectures fulfill the requirements necessary for *continuous availability* – redundancy, extremely fast and reliable failover, and durable data. As shown in Figure 2, an active/active system[11] comprises two or more geographically dispersed nodes that are actively participating in a common application. That is, each node is actively processing and sharing the application load with the other nodes. If a node fails or needs to be brought down for maintenance purposes, transactions or users are switched from the failed or downed node to the surviving nodes, a switch that is undertaken in sub-seconds or seconds. Users already connected to the surviving node are not affected at all.

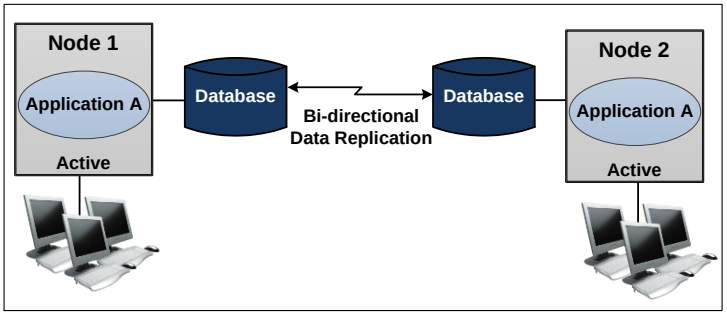

Figure 2 – An Active/Active System

In order for a node to participate in an application, it must have access to an up-to-date copy of the application database. To satisfy the redundancy requirement for high availability or continuous availability, there must be

[9]For more information, see the Gravic white paper, *Choosing a Business Continuity Solution to Match Your Business Availability Requirements*.
[10]If failover is not regularly and successfully tested, fast failover can also be an issue for active/passive systems.
[11]For more information, see the Gravic white paper, *Achieving Century Uptimes with Shadowbase Active/Active Technology*.

at least two geographically distributed database copies in the application network. The database copies may be implemented as network-attached storage or as a database copy directly attached to each processing node, as shown in Figure 2. The database copies are kept synchronized via bi-directional data replication. Whenever a processing node makes a change to its database, that change is immediately replicated to the other database copies in the application network.

Thus, with respect to active/active systems and our requirements for eliminating unplanned and planned downtime:

Redundancy: There are always one or more processing nodes that are geographically dispersed and take over the load of a node that is taken out of service. Each of these processing nodes always has access to at least one synchronized copy of the application database.

Fast and Reliable Failover: With an active/active system, in the event of a node outage, there is, in effect, no failover. No idle node need be brought into service. There is only the rerouting of transactions or the switching of users from the downed node to one or more surviving nodes. Service is returned to the affected users within sub-seconds or seconds. Since all nodes are actively processing transactions, it is known that all are working properly, and failover faults do not occur. Consequently, users or transactions are rerouted from one node to another with little if any risk.

Durable Data: During normal operations, there are at least two copies of the application database in the network. Synchronous replication maintains them in exact synchronization. Asynchronous replication keeps them synchronized to within tens or hundreds of milliseconds.

Eliminating Planned Downtime in Highly Available Systems

The Problem

Most systems today do not take advantage of an active/active architecture. Rather, an independent, idle passive system backs up the production system. Though the passive system may be using a copy of the production database in read-only mode or may be running other applications, it is not running the production applications that are running on the active system. In these situations, there is no attempt to achieve continuous availability. Outages of minutes to hours as the backup system is brought into service following a primary system failure are deemed acceptable.[12]

In less critical applications, there may not even be a backup system. Rather, when an outage occurs, a new system is procured. However, in all of these systems, users generally expect the application to be available to them during the working day or perhaps even on a 24x7 basis. Application services cannot be taken down for maintenance purposes during such times.

For active/passive architectures, there is often a maintenance window during which the system need not be up and running and during which the system is available for upgrade activities. This maintenance window might be over a night or a weekend. For instance, a stock exchange's trading system can be available for upgrades during non-trading hours. But can the upgrade be completed during the time available? Can the upgraded system be thoroughly tested? What if the upgrade fails? Can the original system be restored prior to the end of the maintenance window? Even more critical are systems that must be up 24x7. In these systems, there is no maintenance window. How are upgrades made to a system that is in active use?

Also, once the upgrade is completed, the commissioning of the newly modified system faces the big-bang syndrome of sudden deployment. Does the upgraded system scale well as the load on it increases? Are all of the external interfaces sufficiently tested? (In some cases, there is no capability to test an external interface when the upgrade is underway). Even if confidence in the upgraded system's integrity is demonstrated through extensive testing, there are often just too many subtle failure scenarios to ever believe that all of the bugs are identified and resolved. The ZDM approach mitigates these concerns.

[12]Though this is the restoration time that management expects, the restoration is often further delayed by the time that it takes for management to authorize a failover to the passive system rather than attempting to recover the failed system. This decision is often a difficult call because of the risk that the passive system may not come up.

The Old Way for System Migrations – The Big-Bang Approach

We call the old method of system upgrading the big-bang approach. The system is taken down during the maintenance window, typically at night or during a weekend. If the upgrade impacts the database – for instance, if the database schema is being modified – the first step is to make a full backup copy of the database (or an incremental copy of the current database, saving all of the changes since the last full backup). The new environment is then set up, and the database is reloaded if necessary into the new environment. The new system is tested to the extent that it can be within the maintenance window and given the constraints of the test environment. For instance, external interfaces may not be available for testing; or a full production load may not be able to be tested.

When time is up, the new system is put into service and the users are happy if all goes well. But all too often, migrations do not go well. It is primarily the limited test time, constrained testing procedures, and the complex and difficult fallback requirement that make the big-bang approach so risky.

The New Way for System Migrations – the ZDM Approach

Applying the technique of ZDM to active/passive systems, applications are available during the entire upgrade process, including the switchover to the upgraded environment. Except for new features, users are substantially unaware of the switch. If the new environment experiences problems either immediately after the switchover or at some later time, users return to the original production system with little if any interruption in services and with no loss of data even if new data was generated before the fallback occurred.

The ZDM process for active/passive configurations proceeds as follows. The first three steps are shown in Figure 3.

Step 1: Configure Environment to be Upgraded – Set up and configure the new environment on a redundant system. This system is usually the backup system (Figure 3a). In some cases, the new environment is configured on the production system if a backup system is not available (Figure 3b). In these cases, the production system must have sufficient processing power and disk space; and it must be able to support the facilities being upgraded (for instance, the operating system cannot be upgraded since that would affect current applications as well). The new environment includes the appropriate hardware, operating system version, database management system version, database schema, and application versions. It is a complete system (or environment if on the same system) ready for production.

Step 2: Load Test Database – Load a test database onto the system being upgraded. This database is a special test database designed for system verification, a snapshot of the current production database, the entire current production database or subset thereof acquired via an online copy, or any other database that allows the new system to be thoroughly tested.

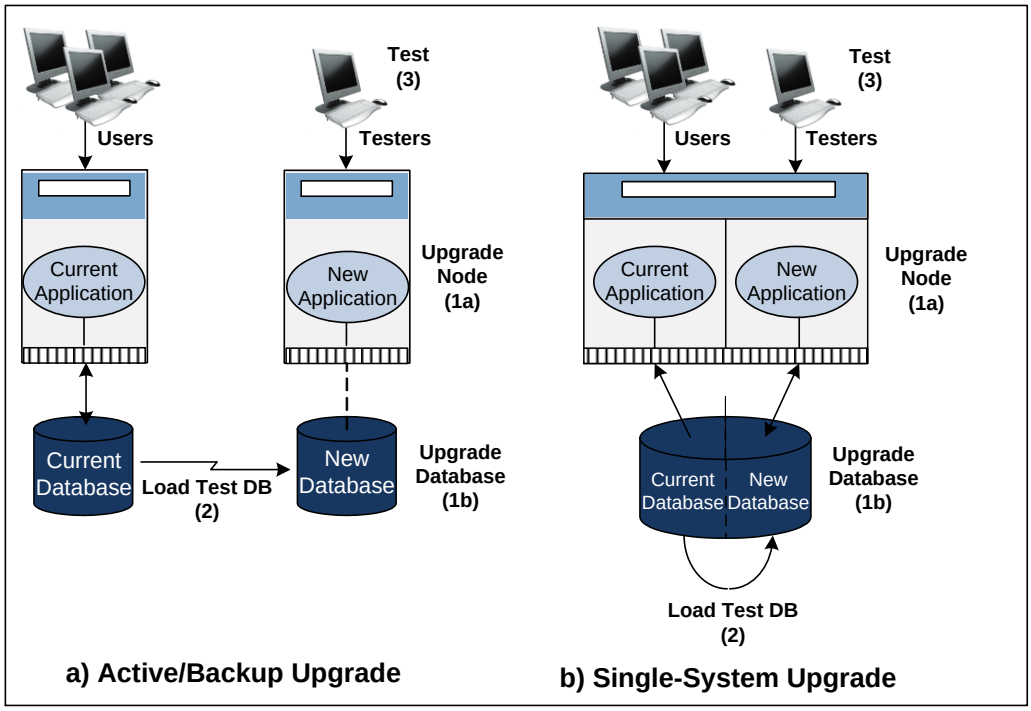

Figure 3 – ZDM for High Availability, Steps 1, 2, 3

Step 3: Test, Test, Test – Test the system as long as needed. This testing can take days, weeks, or even longer. Thoroughly test not only the application logic but all interactions with ancillary systems (which usually support verification transactions to ensure operability). Proper testing must include testing at the full anticipated loads and beyond. During this extended test time, the original production system continues to service the users.

Steps 4, 5, and 6 are shown in Figure 4.

Step 4: Synchronize the Database – When testing is complete, and the upgraded environment certified for use, the preparation for switchover begins. The first step is to synchronize the upgraded environment's database with that of the production system. This synchronization is accomplished with an online load that replicates the production database to the upgraded system while the production system continues in operation. Alternatively, if the test database is a full copy of the production database, and if the upgrade process did not last too long, the production system queues changes made during the test process and drains these changes to the upgraded environment via data replication.[13]

Step 5: Maintain Database Synchronization – When the current production database is loaded onto the upgraded environment, it is now important to keep the newly loaded copy synchronized. This synchronization is done by configuring a data replication engine that replicates all new updates made to the production system to the upgraded environment so that the upgraded environment's database is always an up-to-date copy of the production database.

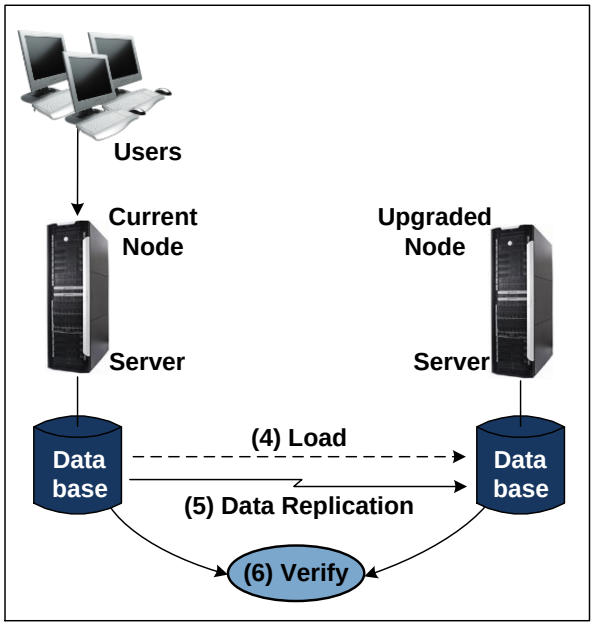

Figure 4 – ZDM for High Availability, Steps 4, 5, 6

[13]The schemas of the two systems may be different if heterogeneous data replication is used.

Step 6: Verify the Database – As an option, it is wise to use one of the available verification and validation utilities to ensure that the upgraded database is indeed a viable and complete copy of the production database, especially if there is a change in the database manager or database schema. If any issues are encountered, the target database is purged and reloaded after fixing the issue. The database is then re-verified before putting it into production. This process is repeated until the target database is correct. During this time, production services are unaffected.

Steps 7, 8, and 9 are shown in Figure 5.

Step 7: Configure Reverse Replication – It is a best practice to make a copy of the production database to be used in the event that the new system corrupts the database due to an undetected bug (Step 7a). To ensure that no data is lost if a fallback is necessary, optionally configure reverse replication so that changes made to the database of the upgraded environment after users are cut over are replicated back to the original production system (Step 7b).

Step 8: Switch Over Users – At this point, the upgraded environment is ready to be put into production. Users are switched over to the upgraded environment either en masse (admittedly a big bang but onto a thoroughly tested system) or, if the database and user groups are logically partitioned, piecemeal by switching over one group of users at a time. If the data replication engine is synchronous, or if it is asynchronous and handles data collisions (the simultaneous updating of the same data object in two different database copies), then users can also be slowly moved over a few at a time to ensure proper operation, without requiring user group or database partitioning. Regardless, switching over users in a controlled fashion allows you to check scaling as the load increases, hopefully avoiding any latent loading issues. When all users are switched over, the upgraded system is fully in production. However, it may be wise to keep the old production system running with its database synchronized via the reverse replication channel so that no data is lost if users must return to the original system due to a fallback.

Step 9: Fall Back If Necessary – If problems appear in the upgraded system either during the cutover process or afterwards, users return to the original production system while the problems in the upgraded environment are corrected. Just as with cutover, fallback is very fast and reliable with no data loss (provided the old production system was kept running with its database synchronized).

Step 10: Upgrade Original Production System – Once the upgraded system is in production long enough to inspire confidence, the original system is decommissioned and upgraded via the same ZDM process. This whole process can be repeated as many times as necessary for all systems requiring the upgrade.

Thus, in terms of the requirements stated earlier, user switchover is fast because users transfer from one operational, known-working system to another. Both failover and fallback are reliable because the users are being switched to a system that is known to be operating correctly, and with synchronized databases. The bottom line is that the system has been upgraded without any application outage to the users. Even in an active/passive system (or in some cases a system with no backup), planned downtime is eliminated or dramatically reduced when compared to other methods.

One important note to make is that depending upon the type of upgrade being made (such as the operating system rather than an application), the operating node may be a single point of failure during certain steps of the upgrade procedure. The reason is because redundancy has been eliminated, and the system is running as a single system. If that system or its network fails, the applications are down. This problem is mitigated by making appropriate backups at key points during the ZDM process (such as at Step 7a), or by using a third system (if available) that acts as a backup to the active node during the upgrade process.

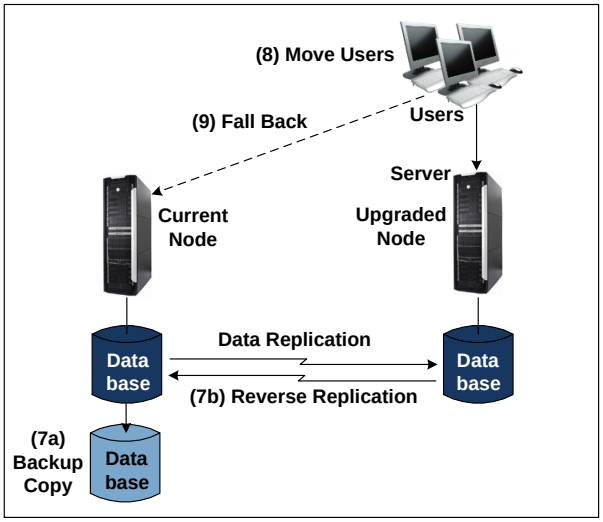

Figure 5 – ZDM for High Availability, Steps 7. 8. 9

Eliminating Planned Downtime in Continuously Available Systems

Multi-node Active/Active Systems

Active/active systems provide continuous availability because they recover quickly from failures in seconds or even in sub-seconds. This same capability is put to use to eliminate planned downtime. Because users migrate so easily and reliably from one processing node to another, nodes are taken down one at a time to be upgraded, allowing an upgrade to be rolled through the application network. While a node is down for maintenance, the other nodes in the system are servicing the users. Figure 6 shows a two-node active/active system before an upgrade is started. The ZDM upgrade process used to eliminate planned downtime in an active/active system comprises the following steps.

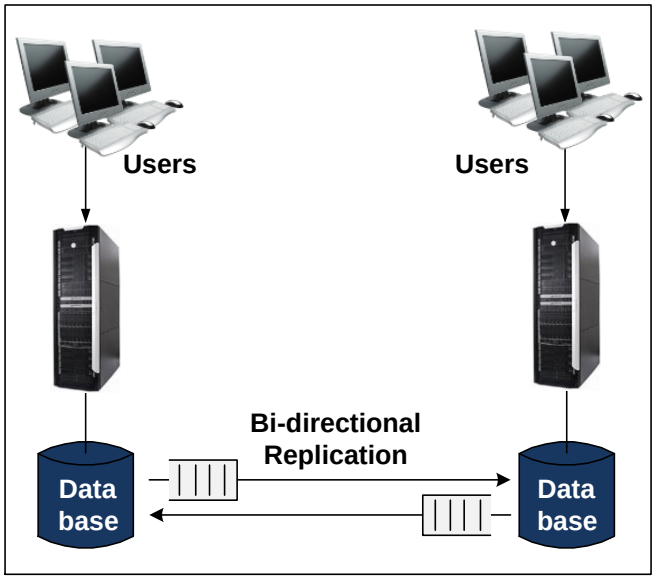

Figure 6 – An Active/Active System to be Upgraded

The first three steps are shown in Figure 7.

Step 1: Take Down the Node to be Upgraded – Move the users from the node to be upgraded to one or more of the other nodes. If dynamic transaction routing is used, reroute all transactions to the other nodes. Stop data replication from and to the node being removed from service, and take down the node. Changes made by other nodes in the application network are queued for subsequent delivery by their replication engines when the upgraded node is returned to service.

Step 2: Upgrade the Downed Node – Perform whatever maintenance is to be undertaken on the downed node. This maintenance might include an application upgrade, the installation of a new operating system version, a new database management system version, or even the migration to a new platform.

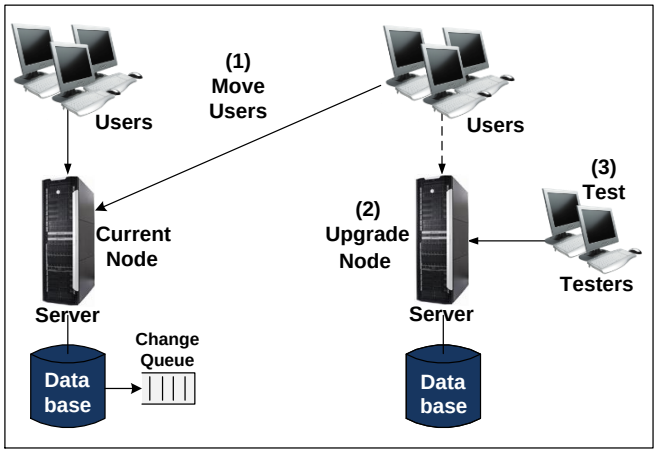

Figure 7 – ZDM for Continuous Availability, Steps 1, 2, 3

Step 3: Test the Upgraded Node – The upgraded node is now thoroughly tested before returning it to service. If the test procedure lasts for an extended period of time, for example, days or even weeks, there is no problem because the surviving nodes in the active/active application network continue to provide full user services.

Steps 4 and 5 are shown in Figure 8.

Step 4: Synchronize the New Database – Once the upgraded node has passed its test, load the current production database onto the database on the upgraded node. The load facility must be capable of faithfully loading the contents of the production database to the target database while the production database is being actively updated. The load facility must also keep the new database synchronized following the load as further updates are made to the production database.

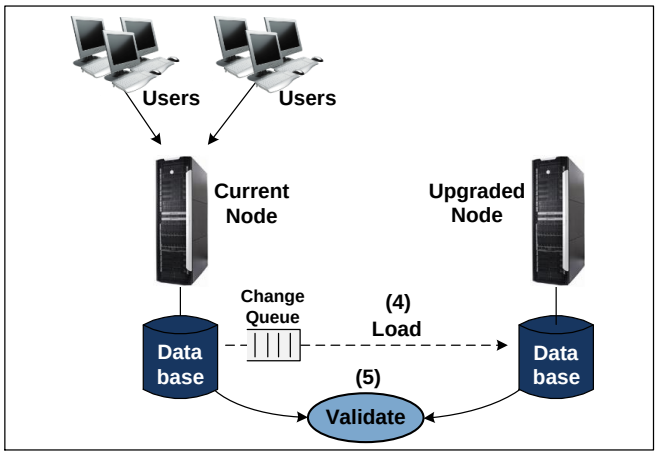

Figure 8 – ZDM for Continuous Availability, Steps 4, 5

Alternatively, the active nodes may have queued changes to the upgraded node. If the overall downtime of the upgraded node was short, or if the amount of data changes queued was small, drain those changes to the upgraded node to resynchronize it with the production database. Regardless of the method used to bring the new database into synchronism with the production database, continue to use data replication to keep the new database synchronized.

Step 5: Validate the New Database – It is good practice to run a verification and validation utility to ensure that the database on the node returning to service is correctly synchronized with the current production database. If any issues are encountered, the target database is purged and reloaded after fixing the issue. The database is then re-verified before putting it into production. This process is repeated until the target database is correct. During this time, production services are unaffected.

Steps 6 and 7 are shown in Figure 9.

Step 6: Make a Backup Copy of the Production Database – Though optional, it is good practice to make a backup copy of the production database at this point, in case the newly upgraded application is incorrect and consequently corrupts the database when the upgraded system is put into production. This backup copy of the database is used to reestablish a known correct database at a point in time.

Step 7: Begin Bi-directional Replication – At this point, the upgraded node is ready to return to service. It is time to begin bi-directional data replication between the production database and that of the upgraded node so that changes made by either node are reflected in each database.

Steps 8 and 9 are shown in Figure 10 and Figure 11.

Step 8: Put the Upgraded Node into Trial Use – Move a small subset of users to the upgraded node. If this movement is successful, move more users to the upgraded node. In this way, the upgraded node is returned to service gradually and in a controlled manner to verify proper scaling and processing.

Step 9: Fall Back if a Problem Occurs – If a problem occurs during the gradual migration of users to the upgraded node, be prepared to revert back to Step 1 to fix the problem. Move the users off the newly upgraded node, stop replication, take down the node, and fix the problem. Follow Steps 2 through 9 to reattempt returning the upgrade node to service.

Step 10: Put into Full Service – When all of the upgraded node's users are returned to it, and when operation is satisfactory, the upgrade of this node is complete. It is returned to full service. Another node can now receive the upgrade. In this way, the upgrade rolls node-by-node through the rest of the application network on whatever timetable you deem appropriate.

One important note to make is that in a two-node system, the operating node is a single point of failure during the upgrade. The reason is because redundancy is eliminated, and the system is running as a single system. If that system or its network fails, the applications are down. Active/active systems using three or more nodes avoid this problem.

Sizzling-Hot-Takeover (SZT)

ZDM, as described above, applies to systems running in an active/active configuration. If your system is not running active/active, it may seem a simple step to provide bi-directional replication between your current primary and backup systems and to put the backup system to work as a cooperating member of an active/active pair.

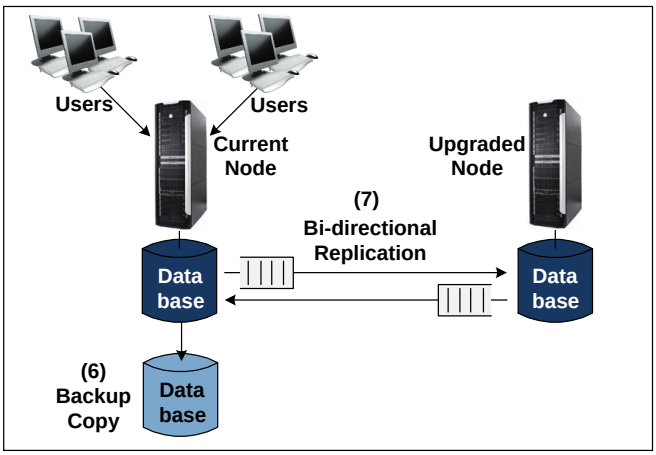

Figure 9 – ZDM for Continuous Availability, Steps 6, 7

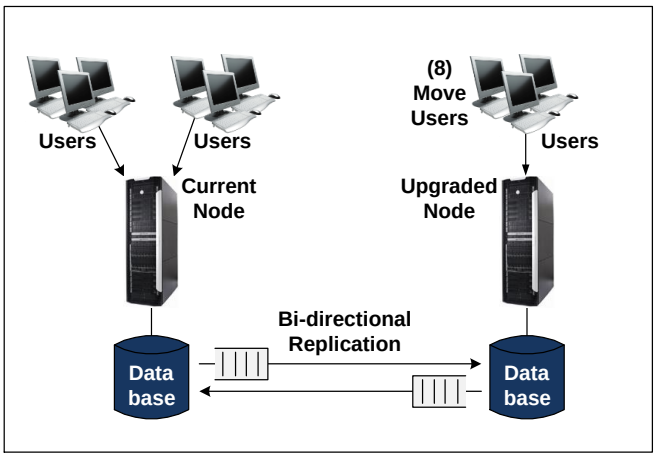

Figure 10 – ZDM for Continuous Availability, Step 8

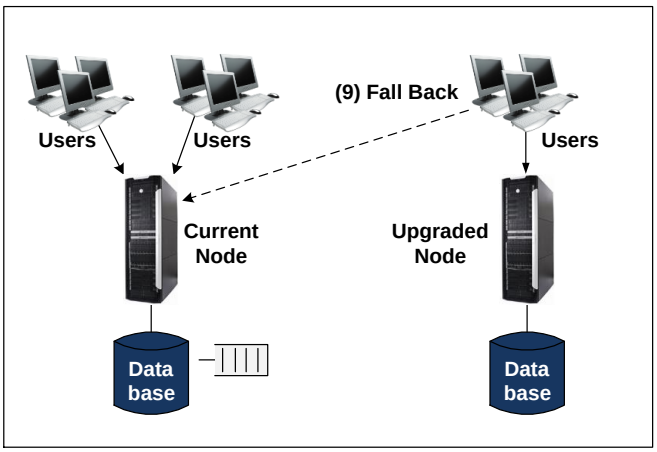

Figure 11 – ZDM for Continuous Availability, Step 9

However, things are not so simple. There are many application structures that may have to be modified in order for the application to be active/active-ready. For instance, data collisions may occur if two nodes try to update the same data item at the same time. Unique numbers such as invoice numbers may no longer be unique across the nodes. Nodes may have to see in-memory context resident in other nodes.

If it is deemed too expensive or impractical to move to full active/active, a partial step is to move to a Sizzling-Hot-Takeover (SZT) system. This system configuration, shown in Figure 12, is the same as that of an active/active system except only one node is processing update transactions that change the database. The applications on the standby node are up and running and may optionally be processing read-only requests (although they do have the local database open for read/write access, ready for takeover if necessary). Data replication keeps the database on the standby system synchronized with that on the active node. To ensure that full end-to-end processing is operational on the standby system, verification test transactions are periodically sent to the standby node's applications.

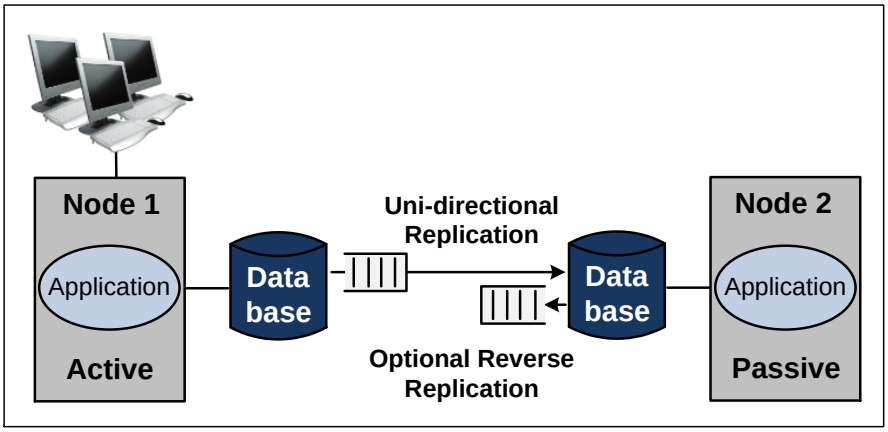

Figure 12 – Sizzling-Hot-Takeover (SZT)

In this way, applications do not have to be modified to run in an active/active, multi-node environment. However, an SZT architecture maintains all of the failover properties of active/active systems so necessary to ZDM. It is known that the standby node is operational – it is easily tested by continuously sending it test or verification transactions. If the primary node fails, users quickly reconnect to the standby system (or transactions are rerouted to it) within seconds. If data replication is configured as bi-directional, the operational node resynchronizes the downed node when the downed node is returned to service.

The active/passive configurations so common in today's IT environments are often extended to a SZT configuration by simply implementing a suitable bi-directional data replication product to keep the active and backup databases in synchronization. In this way, with no application changes and perhaps with little or no additional hardware expense, the benefits of faster recovery in the event of a node failure and of ZDM are achieved.

Eliminating Planned Downtime with HPE Shadowbase Data Replication

The ZDM processes described above require three facilities in order to eliminate planned downtime:

- A data replication engine with low latency and, in certain cases, with bi-directional capability
- An online data load utility
- A data validation and verification utility

Shadowbase software is a complete set of products designed to meet these needs. Of particular importance for ZDM is the HPE Shadowbase data replication engine and its online load facility, SOLV. They are high-volume, high-speed, and reliable components that impose only a small footprint on the systems that use them.

The HPE Shadowbase Data Replication Engine

The HPE Shadowbase engine provides noninvasive data replication from one environment to another.[14] Not only does the HPE Shadowbase engine synchronize homogeneous databases, but it provides heterogeneous synchronization as well. The databases may be in the same or different systems and the systems may use different operating systems or may even be manufactured by different vendors. The databases themselves may also be from different vendors. HPE Shadowbase software supports all of the different ZDM architectures described above, single-system, active/passive, active/active, SZT, with asynchronous and synchronous data replication. So whatever your particular requirements, there is a Shadowbase solution to meet them.

[14]For more information, visit ShadowbaseSoftware.com.

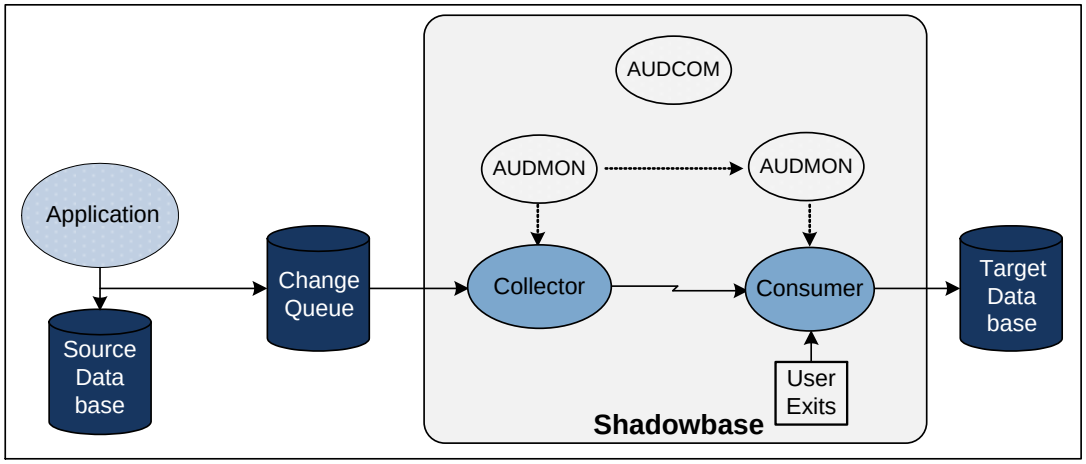

Figure 13 – HPE Shadowbase Data Replication Engine Architecture

Shadowbase replication is driven from a change queue, (Figure 13) which is a persistent record of all changes made to the source database. The change queue may come from any one of a number of sources:

- An audited database under the control of a transaction manager such as the HPE NonStop TMF (the Audit Trail) may generate the change queue.
- An application may generate the change queue.
- Database triggers provided by HPE Shadowbase replication may generate the change queue (called a *Database of Change*, or DOC, file).

The change queue's persistence is important because this persistence is what supports the HPE Shadowbase replication engine's restart and recovery capability in the event of a system or network failure.

Within the HPE Shadowbase replication engine, one or more Collectors read the changes from the change queue and send them via a communication channel to one or more Consumers, which post the changes to the target database. Data transformation, mapping, cleansing, filtering, and scrubbing facilities are provided to perform any required transformation of the source data to the target data (programmatic customization is possible via a user exit mechanism).

The HPE Shadowbase AUDMON process monitors the health of all modules and automatically restarts a module if it fails. The HPE Shadowbase AUDCOM process provides a series of commands and status information for the user to control and configure.

The Benefits of HPE Shadowbase Data Replication

The Shadowbase data replication engine brings many benefits to the synchronization of data in distributed systems, supporting prevention of both planned (ZDM) and unplanned outages.

- ***Flexible –*** It supports active/passive, active/active, and SZT configurations with uni-directional or bi-directional replication.

- ***Persistent and Reliable –*** Its monitor, AUDMON, monitors the health of the Collectors and Consumers and restarts a failed module if necessary. In HPE NonStop servers, AUDMON is a fault-tolerant process pair. On other platforms, AUDMON is a persistent process. When transient faults occur, it recovers automatically and picks up where it left off so that all data is successfully delivered.

- ***Configurable and Controllable –*** Via the AUDCOM command interface.

- ***Noninvasive –*** It is usually driven from the change queue and has no direct involvement with the application. The application is generally unaware that its database is being replicated. If asynchronous replication is used, replication does not affect application performance. However, synchronous

replication may increase application response time as the completion of the transaction across the application network is awaited.[15]

- ***Low Latency –*** That is, the time from a change on the source database to the time of that change on the target database is minimal. It is configured to contain no disk-based queue points – data replication is strictly process-to-process.

- ***Minimal Data Loss –***With asynchronous replication, because of its low latency, there is *minimal data loss* in the event of a total source-system failure.[16] With synchronous replication, no data is lost.[17]

- ***Low Overhead –*** It imposes only a small footprint on both the source and the target systems. Not only is resource consumption minimal, but efficient communication buffering also minimizes network utilization.

- ***High Performance –*** It is *multithreaded* for demanding replication environments. Doing so allows database changes to propagate to the target database over multiple paths in parallel.

- ***Scalable –*** Replication of critical processing modules (such as Collectors and Consumers), with balancing of workload between them, ensures that Shadowbase software can scale to meet the most demanding throughput requirements.

- ***Referential Integrity –*** Even in a multithreaded configuration, it is configured to guarantee *referential integrity*. It guarantees the proper serialization of all changes being applied to the target database so that transactions are applied to the target in the same order as they were made to the source database.

- ***Heterogeneous –*** Not only can the data structures be different on the source and target databases, but the databases themselves can also be different. Even the operating systems and the platforms on which they reside can be different.[18]

- ***Bi-directional –*** It simultaneously replicates changes being made in either database to the other. Its patented technology[19] protects against ping-ponging, or the return of a replicated change back to the source.

- ***Data Collision Avoidance or Resolution –*** For active/active architectures, its low latency minimizes the chance that *data collisions* occur (the nearly simultaneous updating of the same database row at different copies of the database). If data collisions do occur, it detects and reports them. In many cases, it may automatically resolve data collisions by using its collision-resolution facilities. For other cases, it supports embedding business logic into the replication engine to implement special application-based rules for collision resolution.

- ***Patented Technology –*** It uses innovative and novel techniques for data synchronization and has been awarded several patents.[20]

- ***Synchronous Replication –*** Via the ***HPE Shadowbase ZDL*** and ***HPE Shadowbase ZDL+*** facilities, it may be configured for *synchronous replication*. These new features guarantee that no change is made to a database unless that change is made atomically to all database copies in the network. Synchronous replication not only eliminates data collisions (***HPE Shadowbase ZDL+***), but it also guarantees that no changes are lost in the replication pipeline in the event of a failure (***HPE Shadowbase ZDL*** and ***HPE Shadowbase ZDL+***).[21]

[15]Shadowbase synchronous replication algorithms use techniques to minimize this application latency.
[16]When using asynchronous replication, changes are replicated from a change queue to the target system. If the source node fails, any changes still in the change queue will not be replicated and will be lost.
[17]Contact Gravic for the availability of this feature.
[18]See the Shadowbase website for a current list of supported databases and platforms.
[19]Strickler, G., et al., "*Bi-directional database replication scheme for controlling ping-ponging*," United States Patent 6,122,630; September 19, 2000.
[20]See Gravic.com/graviclabs/patents/index.
[21]Contact Gravic for the availability of these features.

SOLV, the HPE Shadowbase Online-Load Facility

To support initial target database loading for ZDM, the HPE Shadowbase SOLV facility creates a synchronized copy of the source database at the upgraded target database. It creates the copy while the source database is undergoing active transaction processing, thus avoiding an outage in order to build a target database.[22]

SOLV efficiently moves the source database to the target database. At the same time, SOLV cooperates with HPE Shadowbase replication to ensure that the data which it is loading is properly serialized with the data being replicated. Thus, at the end of the load cycle, the target database is up-to-date. A change queue does not have to be stored and replayed later once the load has completed in order to synchronize the target database, as is required by other load or ETL (extract, transform, and load) utilities.

HPE Shadowbase Compare

Shadowbase Compare compares one database against another even while both databases are being updated and will *validate* that they are the same. Shadowbase Compare performs verification by moving data blocks similar to how SOLV loading does when it performs a load, but instead of writing data blocks to an empty target database, Shadowbase Compare compares them to the data blocks in the target system. Differences that Shadowbase Compare determines to not be in-flight updates are reported for further action. If there are no differences, the databases are validated. This function is often referred to as *database compare*.

Taking this function a step further, a future enhancement will allow a target database to be re-*synchronized* with a source database by repairing mismatches. If a target row disagrees with a source row, Shadowbase Resync overwrites the target row with the source row. If a target row does not exist, Shadowbase Resync inserts the source row. If a target row exists but is absent in the source, Shadowbase Resync deletes the target row. This function is often referred to as *database repair*.

The Benefits of SOLV, HPE Shadowbase Compare, and HPE Shadowbase Resync

All of these Shadowbase loading, comparison, and resynchronization facilities support HPE Shadowbase replication's fundamentals of heterogeneity, high performance, reliability, non-invasiveness, and low overhead. In addition, they provide the following benefits:

- They do not require the source application or database activity to be paused during the load process. The source application is fully functional and modifying the source database while Shadowbase replication is loading the target database.
- Target data transformations are done in only one place. The same transformations are used for loading, replication, verification, and validation. There is no requirement to repeat the data transformation logic in a bulk extract, transformation, and load operation, which would otherwise be necessary if another tool were used.
- Auditing is generally not required on either the source or target databases.
- Data replication is configured to occur concurrently with the data load or compare operations so that there is no long queue of changes stored and replayed after the load or compare completes.
- During a load operation, that part of the target database that SOLV loaded is immediately in a consistent state and maintains its consistency. The target database may be used for a compare or for target application processing, if appropriate.
- Data compression or encryption may be used across the network between source and target systems.

HPE Shadowbase replication does not require a snapshot to be taken and written to disk. It does not use any intermediate storage of the data to be loaded or compared.

[22] P. J. Holenstein, B. D. Holenstein, G. E. Strickler, "*Synchronization of a target database with a source database during database replication*," United States Patent 7,321,904; January 22, 2008.

Customer Case Studies

The following case studies are of projects in which the customer used the HPE Shadowbase suite of products (built by Gravic, sold by HPE) to undertake an important and successful migration using the ZDM technique.

Casino Administration

A large Atlantic City, New Jersey, casino was using older HPE NonStop servers to manage all aspects of the casino's business. As part of a major upgrade, it not only developed new applications but also had to restructure its databases. The original databases included both Enscribe non-relational structured files and SQL relational tables. The new databases used SQL exclusively.

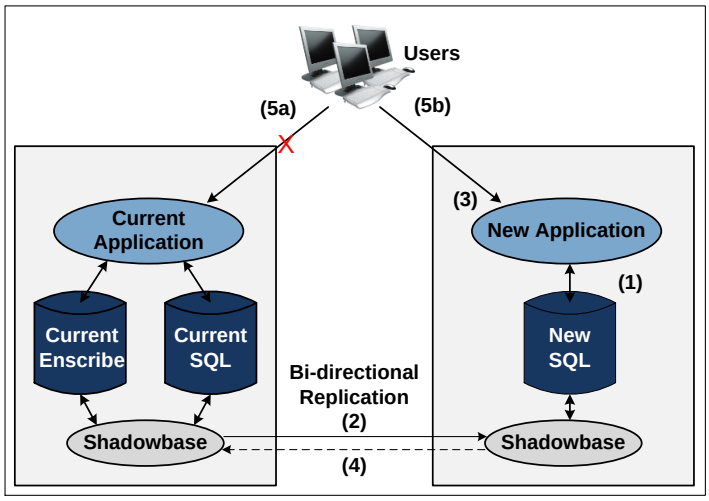

Figure 14 – Casino Administration

Since casino activity is 24/7 (the casino operations never shut down), the casino had to bring up and place into operation the new applications and databases without impacting operations. The casino accomplished this procedure by using the HPE Shadowbase Zero Downtime Migration (ZDM) capability (Figure 14). The casino first brought up the new applications on a separate system (1) and used Shadowbase ZDM to load the online database to the new system. Data replication kept the new database in synchronism with the operational database (2). Shadowbase ZDM also provided the schema translation between the Enscribe files and the new SQL tables.

The new applications were started and tested thoroughly (3). Once satisfied that the applications performed properly, the casino was ready to put the new system into operation. It first took a full backup of the current operational system (as is best practice) just in case it was needed later. The casino then initiated reverse replication (4) so that changes made to either system were reflected in the other system. This replication allowed the casino to phase users over slowly. Reverse replication also provided a fallback safety net in case the new system demonstrated problems.

The casino then began phasing over users to the new system (5a, 5b) without denying service to any of the users. The old database was kept in synchronism with the new database for a period of time. In this way, users could return to the old system in the event of a problem in the new applications. Once the new system had proven itself, the casino shut down the old system.

Paper Manufacturer

A paper manufacturer had its applications on an HPE NonStop server. It wished to move some of the applications to a Windows SQL Server environment (Figure 15); but since it could not take its operations offline while it did so, it used Shadowbase ZDM technology to achieve this migration.

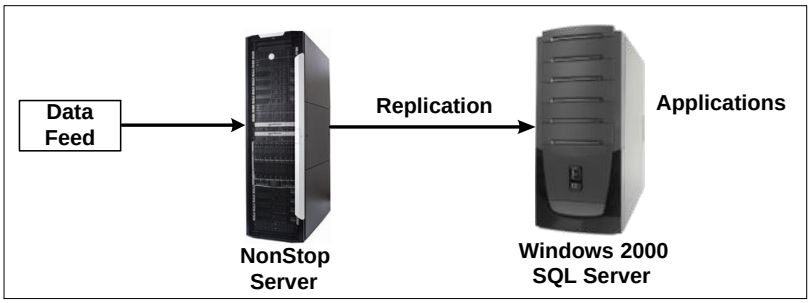

Figure 15 – Paper Manufacturer

Though some of the applications were moved to the SQL Server system, certain application functions dealing with processing the incoming data feed needed to remain on the NonStop server. Thus, the NonStop server continued to receive and process the data feed. It then trickle-fed the results to the SQL Server system via HPE Shadowbase asynchronous data replication.

Master/Slave Cell Phone Application

A cell phone application implemented as an active/active master/slave architecture uses a *designated winner* algorithm[23] to resolve collisions. As incoming transactions make updates to a slave node's database, the updates are replicated to the master node. The master node resolves any data collisions, and then replicates in parallel the accepted updates to all other nodes, including the node that initiated the update (Figure 16). Since the master node rejects a slave node's update due to a data collision, the slave node's database is corrected with the winning update.

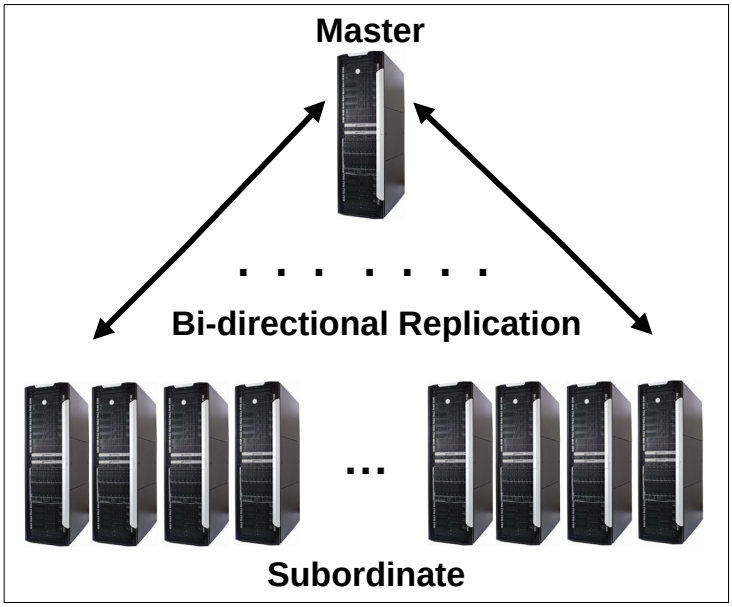

Figure 16 – Master/Slave Cell Phone Application

The application makes use of the master/slave architecture and HPE Shadowbase data replication to perform node upgrades without denying service to any of its users. When a slave node requires upgrading, it is removed from service. Further user requests are simply distributed among the remaining slaves. The slave node is then upgraded, tested, and returned to service by letting the master node know that it is now operational. Each slave node in the application network repeats this procedure.

In order to upgrade the master node, one of the slave nodes must first be promoted to master by reconfiguring its role. The old master is then upgraded and returned to service as a slave node. If desired, it is promoted back to being the master node. Additional slave nodes are easily added or removed for capacity adjustment by simply notifying the master node of the slave node's addition or removal.

The Login Request Complex for a Major Internet Service Provider

A major Internet Service Provider (ISP) serves millions of users worldwide. At any one time, several million of these users may be logged on. Continuity of service is mandatory.[24]

Originally, a sixteen-server Linux/Sybase Login Request Complex backed up by an additional sixteen servers handled login requests (Figure 17). This complex had reached the limits of its capacity. Expanding it would not only be very costly, but the sixteen independent databases created a system management nightmare. The ISP decided to move its Login Request Complex to a four-processor, HPE NonStop active/active system. It had to do this move with minimal impact to its customer base. The ISP chose the HPE Shadowbase replication engine for this purpose because of its rich feature set, enabling them to perform the complex migration.

At this point, the NonStop system was ready to be put into production. At first, only read requests were routed to the NonStop system. Update requests to modify existing user profiles were still routed to the Linux/Sybase systems, with the changes being replicated to the NonStop system by HPE Shadowbase replication. After a period of satisfactory performance, new users were assigned to the NonStop system, which handled both read requests and update requests for these users. Finally, all user login requests, both read and update, were routed to the NonStop system. This routing allowed a phased cutover of users so that the load on the new complex could be slowly increased. The entire Linux/Sybase Login Request Complex and Change Capture Complex were then retired.

The migration proceeded cautiously over a period of months. Several hundred million user accounts were migrated to the NonStop system with no impact on user service. The new NonStop active/active system not only provides four copies of a unified and reliable database of all user accounts, but it is also easily expandable by adding nodes to the active/active system.

[23]Holenstein, B. D., et al, "*High availability designated winner data replication*," *U.S. Patent No*. 7,523,110; April 21, 2009.
[24]For more information, see the Gravic case study, *HPE Shadowbase Helps a Major ISP Migrate from Sybase to HPE NonStop with No Downtime*.

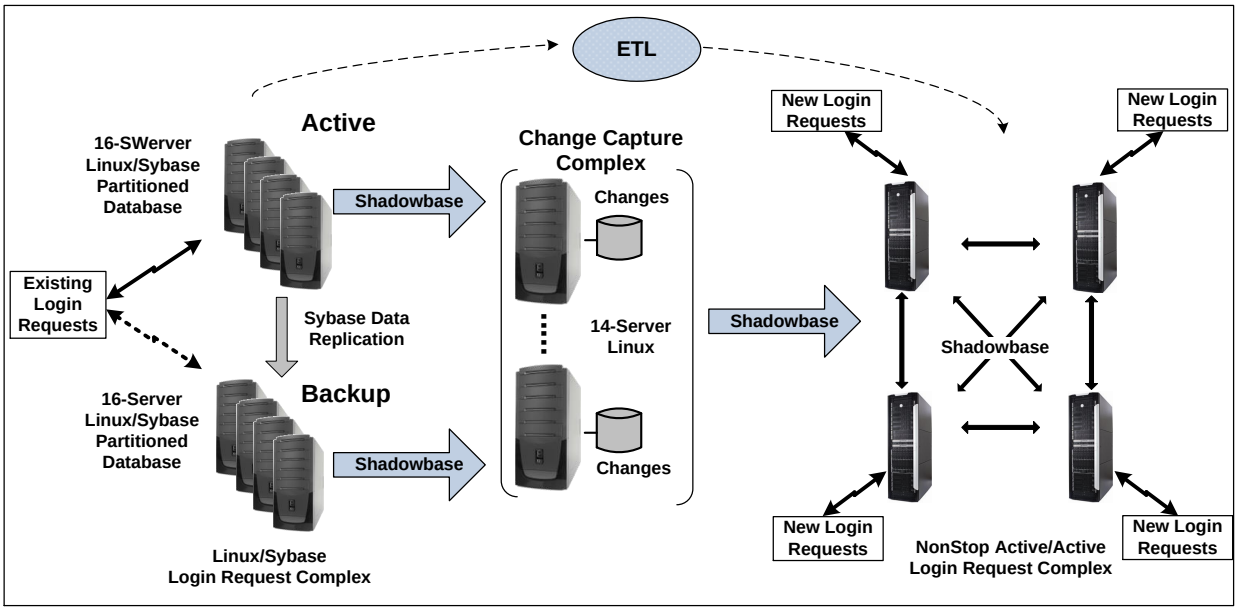

Figure 17 – Login Request Complex for Major ISP

This migration was large and tremendously complex, but because of the capabilities of the HPE Shadowbase product suite, it was accomplished without any loss of service to the ISPs users. It represents a very powerful demonstration of the use of Shadowbase architecture to achieve ZDM.

Summary

System upgrades in conventional IT environments are expensive and can lead to extensive system downtime. For standalone systems, the system must be taken down, upgraded, and returned to service during a maintenance window. Upgrading may take a long time, and an upgrade gone badly could prevent a system from being returned to service in the required time.

Even if a backup system is available and is used for the upgrade, it is necessary to switch operations from the primary system to the newly upgraded system within the maintenance window. Switching over to the upgraded system could be lengthy, during which time both systems are down. The switchover may also fail, following which neither system may be functional.

With active/active systems and their close cousins, SZT systems, failover is virtually instantaneous and fault-free. With these systems, planned downtime (as well as unplanned downtime) is eliminated, leading to exceptionally high availabilities.

The same techniques used by active/active systems to eliminate planned downtime are minimized or even eliminated in active/passive configurations by deploying data replication technology and by using the ZDM technique.

The HPE Shadowbase suite of products (built by Gravic, sold by HPE), provide the facilities needed for zero downtime migration, for standalone, active/passive, SZT, and active/active system configurations. These facilities include the HPE Shadowbase data replication engine, the SOLV online-load facility, and the SOLV verification and validation utility. Taken together, these products offer the means to eliminate planned, as well as unplanned, downtime, enabling system upgrades to be performed safely with no loss of business services.

International Partner Information

Global

Hewlett Packard Enterprise
6280 America Center Drive
San Jose, CA 95002
USA
Tel: +1.800.607.3567
www.hpe.com

Japan

High Availability Systems Co. Ltd
MS Shibaura Bldg.
4-13-23 Shibaura
Minato-ku, Tokyo 108-0023
Japan
Tel: +81 3 5730 8870
Fax: +81 3 5730 8629
www.ha-sys.co.jp

Gravic, Inc. Contact Information

17 General Warren Blvd.
Malvern, PA 19355-1245
USA
Tel: +1.610.647.6250
Fax: +1.610.647.7958
www.shadowbasesoftware.com
Email Sales: shadowbase@gravic.com
Email Support: sbsupport@gravic.com

Hewlett Packard Enterprise Business Partner Information

Hewlett Packard Enterprise directly sells and supports Shadowbase Solutions under the name ***HPE Shadowbase***. For more information, please contact your local HPE account team or visit our website.