the Availability Digest

Shadowbase – The Active/Active Solution

March 2007

Active/active systems achieve their extreme availabilities through distributed processing. Multiple independent nodes share the transaction processing load such that any transaction can be routed to any node and be processed properly. Should a node fail, all subsequent transactions are routed to the surviving nodes. Transaction rerouting can be accomplished so quickly that users are unaware of the failure. In effect, there has been no failure; and the perceived availability of the system can be measured in centuries.

The proper implementation of an active/active system requires that multiple geographically-distributed database copies be kept in synchronization so that any processing node in the application network has access to at least two database copies should one fail. Proper database synchronization requires the ability to

- replicate data changes from one database to the other database copies in the application network so that all database copies maintain the same application state,
- copy a database that is being actively updated in order to create or recover a remote database copy,
- compare two databases to verify that they are identical, and
- bring two databases into synchronism if necessary.

The Shadowbase suite of data replication tools from Gravic, Inc., performs all of the above functions. In addition to active/active systems, these products have many other uses, such as providing a hot standby; integrating disparate systems in heterogeneous applications; offloading query, backup, and extract activities from one system to another; online restoration of corrupted databases; and eliminating planned downtime.

The Shadowbase suite of products includes:

- the Shadowbase data replication engine, and
- SOLV, the Shadowbase online copy, verification, and validation utility.

The Shadowbase Data Replication Engine

The Shadowbase data replication engine[1] replicates changes from one database to another in highly heterogeneous configurations. It is an asynchronous replication engine and therefore is transparent to application processing. It imposes no performance impact on application processing and is non-intrusive in that it requires no changes to be made to the applications themselves.

Shadowbase can be configured for unidirectional replication for applications such as hot standby or query offloading. It can also be configured for bidirectional replication to support active/active architectures.

It provides very fast replication and minimizes replication latency (the time from when a change is made to the source database to when it is applied to the target database) by eliminating disk queuing points. Its throughput can be expanded by running it in a multithreaded configuration. In this configuration, it ensures the proper ordering of transactions being applied to the target database to maintain referential integrity.

In its bidirectional configuration, Shadowbase prevents ping-ponging (the rereplication of a change from the target system back to the source system). A major issue with bidirectional asynchronous replication is data collisions. A data collision occurs when nearly simultaneous changes are made to the same row in different database copies such that the replicated changes will overwrite the original changes. Shadowbase provides data-collision detection and supports many strategies for automatic collision resolution.

Heterogeneity

The Shadowbase data replication engine replicates data between a variety of databases running on a number of different systems. Any valid source database running on any valid source system can replicate to any valid target database running on any valid target system, thus supporting a wide range of heterogeneous operations. The current valid source and target systems are tabulated below. Note that any database or system that can act as a source system can also act as a target system.

Source/Target Databases	Source/Target Systems		Target Databases	Target Systems
NonStop SQL	NonStop server		DB2	OpenVMS
NonStop Enscribe	Linux		Sybase	AS400
Oracle	Unix		MySQL	
SQL Server	Windows			

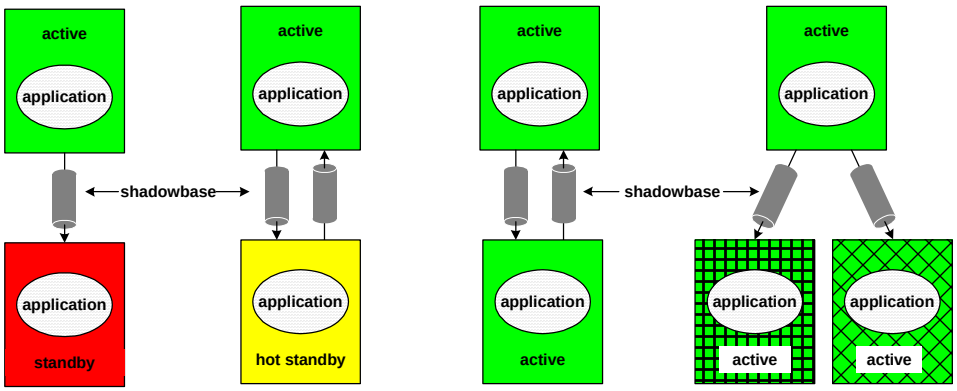
Topologies

A Shadowbase network can consist of any number of heterogeneous nodes connected unidirectionally or bidirectionally in any topology desired. Typical of these topologies are:

- *Active/Standby*, in which an active database is replicated to a standby system which can take over processing should the active system fail. In these cases, the standby applications are generally not active and must be started prior to takeover.

[1] Shadowbase is described in more detail in Chapter 11, Shadowbase, *Breaking the Availability Barrier: Achieving Century Uptimes with Active/Active Systems*, AuthorHouse; 2007.

- *Active/Hot Standby*, in which both systems are fully operational, but only one system (the active system) is handling the processing load. Should the active system fail, the standby can take over processing instantly (often called a *sizzling hot standby* configuration). By using bidirectional replication, failover testing is simple and virtually risk-free since the backup system is fully operational and can have test or verification transactions submitted to it at any time to verify operability. When a failure of the active system occurs, all that is needed is to switch the transaction stream to the hot standby. This system will then become the active system and will keep the alternate system synchronized so that processing load can be switched back to it if desired.



Some Shadowbase Topologies

- *Active/Active*, in which database copies at two or more nodes are kept in synchronism via bidirectional replication, and all nodes are participating in processing the transaction stream.

- *Database Distribution*, in which an active database is distributed to other nodes for query processing, data warehousing, or to support other processing activities. In this topology, the systems are often heterogeneous (e.g., NonStop server transaction processing system to Windows and Linux query processors).

Architecture

Shadowbase uses two different architectures – one for NonStop systems and one for the other systems (Windows, Linux, Unix, OpenVMS, and AS400).

NonStop Systems

In NonStop systems, Shadowbase obtains changes made to the source database from the NonStop server's audit trail. The audit trail contains all transaction information, including transaction boundaries and the before and after images for all updates, inserts, and deletes.

This transaction data is read from the source audit trail by a Collector process. It is buffered, compressed and sent to the target system either via NonStop's Expand protocol or via a TCP/IP session. At the target system, the changes are received by a Consumer process, which will unblock the changes and apply them to the target database. Transaction order is maintained to ensure referential integrity.

One significant advantage of Shadowbase's NonStop architecture is that replication is strictly process-to-process. There are no intermediate disk-queuing points. Shadowbase utilizes the

NonStop audit trail to ensure recovery of transactional data following a node failure. Thus, replication is very fast and leads to minimal replication latency times (and consequently, reduced data collisions and reduced data loss following a node failure).

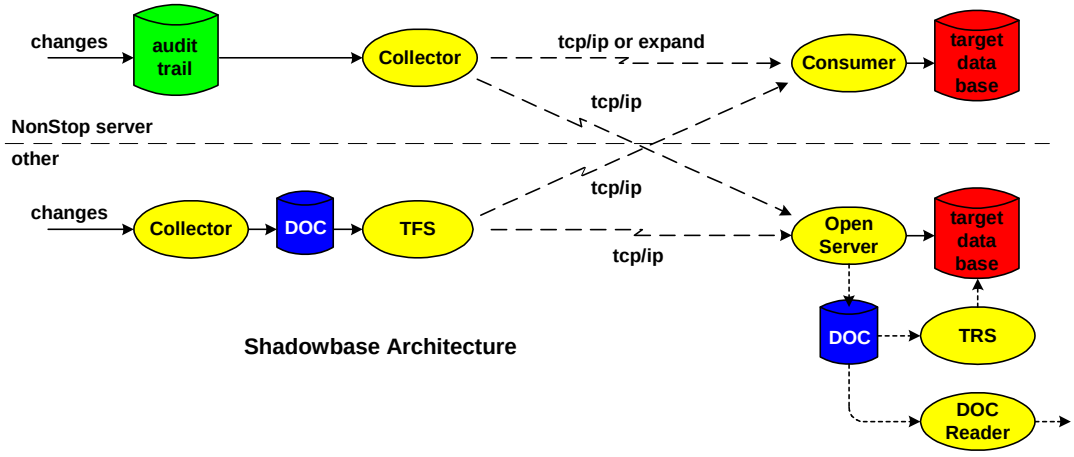
Other Systems

For Windows, Linux, Unix, OpenVMS, and AS400 systems, the Shadowbase architecture is somewhat different. Changes are received from the application by a Collector. In this case, changes could be generated by the application, by a library bound into the application, or by database triggers.

To provide recovery services in the event of a node failure, these changes are written by the Collector to a persistent Database of Change (DOC). From the DOC, a Transaction Forwarding Server (TFS) reads the changes, buffers and compresses them, and sends them to the target system via a TCP/IP session. There the changes are received by an Open Server process, which unblocks them and writes them to the target database.

As an option, the Open Server could instead write the changes to a target-side DOC. This might be used, for instance, to filter aborted transactions. If a target-side DOC is used, a Transaction Replay Server (TRS) reads committed transactions from the DOC and applies them to the target database. In addition, a DOC Reader is provided; it can send changes to other application processes.

In this architecture, the source system can be any one of the source systems supported by Shadowbase. Likewise, the target system can be any one of the target systems supported by Shadowbase.



Shadowbase Architecture

Mixed Systems

Shadowbase provides for NonStop systems to act as either source systems or as target systems for any other supported system. If replication is to be from a NonStop server to another type of system, changes made to the NonStop database are sent by the NonStop Collector to the Open Server on the target system.

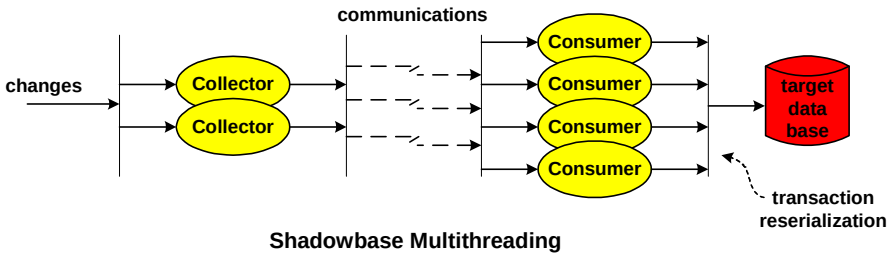
If replication is to be from another type of system to a NonStop server, data changes are sent by the source TFS process to the NonStop server's Consumer process.

In this way, Shadowbase provides complete heterogeneity between any supported systems and databases.

Multithreading

The three critical threads in Shadowbase may be multithreaded to increase replication throughput. These threads are the Collector, the communication channel, and the Consumer. Each can have its own set of threads independent of the others.

Multiple Collectors can read changes made to the source database and can queue them independently to the communication channels. Multiple communication channels may be used to transfer blocks of changes from the source system to the target system. Multiple Appliers (e.g. Consumers or Transaction Replay Servers) will receive communication blocks and will apply the changes to the target database.



Shadowbase Multithreading

If related transactions can flow over multiple threads, one must be careful to apply the transactions to the target database in the same order that they were applied to the source database in order to ensure the referential integrity of the target database. Shadowbase ensures proper order by reserializing the transactions before they are applied to the target database.

If a target-side DOC is used, transactions may be held in the DOC until they have committed. The entire transaction can then be read from the DOC and applied to the target database. This has the problem, however, of bunching transactions, which will cause load peaks at the target database that are not seen at the source database.

Data Transformation

In heterogeneous environments, the database structures of the source and target systems are almost certain to be different. Even in homogeneous environments, data structures and field definitions may be different. An extreme case is replication between a relational database and a file system.

Therefore, it is important that the data replication engine be able to reformat changes as they move from the source system to the target system. Shadowbase provides two facilities for doing this:

- Data format changes can be specified via Shadowbase's Transformation and Mapping Facility, which provides a scripting language for specifying data transformations.

- Most of the Shadowbase components support embedding user exits, which are customer written transformation algorithms for specifying to Shadowbase how the data is to be filtered, cleansed, or transformed. These components include the Consumer, the Open Server, the DOC reader, and the TFS and TRS processes.

Management

The management of the Shadowbase environment is provided by several facilities:

- AUDMON monitors the Shadowbase components in a node and will automatically restart a critical process that has failed. In NonStop servers, AUDMON is implemented as a fault-tolerant process pair.

- AUDCOM provides a command interface so that users can configure and control Shadowbase and monitor its status. AUDCOM may run on either the source system or the target system.

- The Shadowbase Enterprise Manager (SEM) is a Windows GUI that provides integrated command and monitoring support for the Shadowbase components running on the various platforms.

SOLV

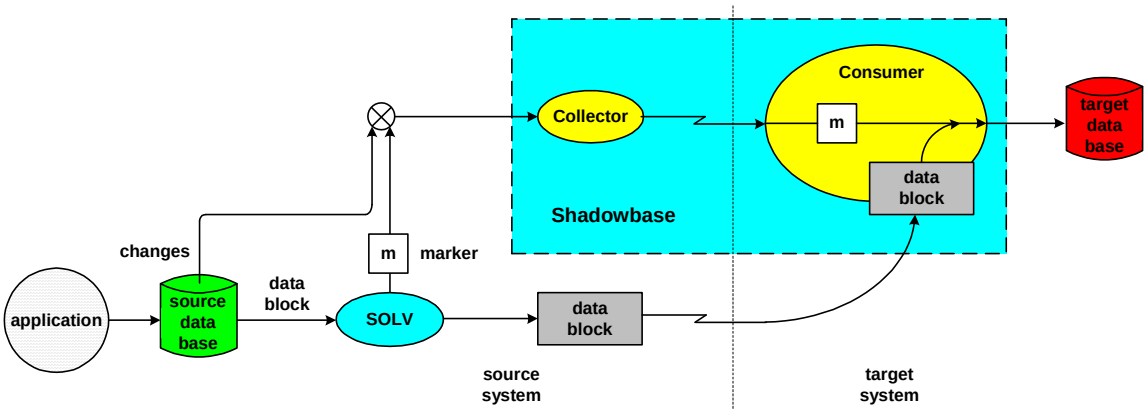
The Shadowbase Online Copy, Verification and Validation utility (SOLV)[2] provides the facility for the online copying of databases, for verifying that two databases match, and for correcting a database that is out of synchronism.

Online Copying

In active/active systems, it is imperative to be able to create a copy of an active database for a node that is to be brought into service, whether it is being repaired or it is a new system node being introduced into the network. The source node cannot be taken down while the copy is being made since it is in active service.

A common way to do this is to take a snapshot of the database at a given point in time. However, the snapshot may take several hours to transfer and load into the target database; and it is then a copy of the database that is several hours stale. To bring it up to date, the changes that have queued during the snapshot interval must be replayed, a process that itself can take several hours – perhaps longer than the snapshot took.

SOLV makes a copy of an active database that it keeps up-to-date as the copy proceeds. Thus, when the copy is complete, the target database is immediately ready to use. SOLV accomplishes this by integrating closely with the Shadowbase data replication engine.



Online Database Copy with SOLV

[2] SOLV is described in more detail in Chapter 12, SOLV, *Breaking the Availability Barrier: Achieving Century Uptimes with Active/Active Systems*, AuthorHouse; 2007.
Holenstein, P. J., Holenstein, B. D., Strickler, G. E., *Synchronization of Plural Databases in a Database Replication System*, United States Patents 6,745,209 and 7,003,531; June 1, 2004 and February 21, 2006.
Contact Gravic, Inc., (www.gravic.com) for the availability of specific features of SOLV.

SOLV is the copy process. It will lock and read a block of data from the source database and will send it to the Consumer, which will write it to the target database. To keep the target database up-to-date, changes to the source database are also replicated by Shadowbase to the target database, as described previously.

However, the flow of data blocks and the flow of changes must be synchronized to ensure that the data block has been written to the target database before subsequent changes to that data block are applied. To accomplish this, SOLV writes into the data stream a marker indicating that a specified block has just been transferred. When the Consumer receives the marker, it will insert the referenced data block into the replication stream at the place of the marker. Thus, any data block changes that were received after the data block was read from the source system are guaranteed to arrive only after the data block has been written to the target system.

SOLV can be multithreaded to improve copy performance. It can also be throttled to prevent over-utilization of system resources.

Verification and Validation

A target database can be verified by a simple extension to SOLV. Rather than writing a data block to the target database, that block is instead read from the target database and compared to the block received from the source system. Comparison can be based on comparing the contents of the source and target rows or by simply comparing checksums or row-version indicators.

This procedure protects against row differences due to replication latency (i.e., inflight changes that have not yet reached the target database). The rows are guaranteed to be identical since the source rows were locked up to the point of the marker. If there is a difference in the rows, the error is reported.

Database Resynchronization

Verification and validation can be extended to resynchronize the databases if there is a comparison error. This is done by using the source-row contents to repair the target row.

If the rows mismatch, the target row is replaced with the source row. If the source row exists but the target row does not, the source row is inserted. If the target row exists but the source row does not, the target row is deleted.

Zero Downtime Migration with Shadowbase and SOLV

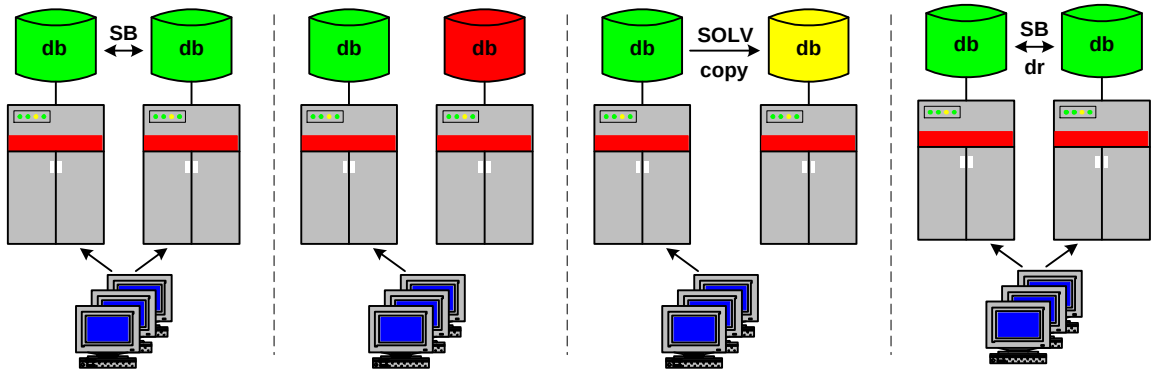
Shadowbase, in conjunction with SOLV, can be used to eliminate planned downtime due to upgrades or migrations. Using an active/active system as an example, the node to be upgraded is taken out of the application network. During this time, the remaining nodes in the application network handle the entire transaction load. In the following figure, the green databases are active, the red database is offline, and the yellow database is being loaded.

The decommissioned node is upgraded and tested. When it is ready to be returned to service, SOLV is used to recreate its application database. Shadowbase replication then keeps that database current while the node is returned to service.

Bidirectional replication is turned on at the upgraded node, and transaction flow is restored to it. Normal operation has now been restored. However, should the upgraded node exhibit problems, it is straightforward to take it once again out of service, correct the problem, and return it to service with no disruption to the services being provided to the users.

If other nodes are to be upgraded, the upgrade process can be rolled through the application network one node at a time.

This same process can be used to add new nodes to an application network. It can also be used to upgrade a single system if another system is available on loan or if the system can be partitioned to support two instances of the application.



Zero Downtime Migration with Shadowbase and SOLV

The Future

One serious need which is not yet available for active/active systems is efficient synchronous replication. With synchronous replication, there are no data collisions; nor is there any data loss following a node failure. However, current methods for synchronously replicating data impose a serious penalty on the performance of their supported applications.

Gravic plans to introduce an efficient synchronous replication product based on their patented coordinated commit method. The coordinated commit method uses asynchronous replication to propagate changes to the target system but coordinates the commits of transactions among the various nodes in the application network after all changes have been replicated.[3]

Gravic

Shadowbase and SOLV are products of Gravic, Inc., of Malvern, Pennsylvania, USA (www.gravic.com). Gravic is the result of the merger of ITI, Inc., the developer of the Shadowbase product line and Compucon Services Corporation (CSC), a turnkey custom software development house.

Today, the Shadowbase Products Group (SPG) develops and supports the Shadowbase line of products.

Gravic has accumulated significant patent coverage for its products. In the Shadowbase line, these patents cover asynchronous replication, synchronous replication using coordinated commits, collision avoidance, and the prevention of ping-ponging, among other inventions.

[3] Coordinated commits are described in Chapter 4, Synchronous Replication, *Breaking the Availability Barrier: Survivable Systems for Enterprise Computing*, AuthorHouse; 2004.