Business Forces Driving the Need for Continuous Availability

While the pace of business continues to increase, tolerance for downtime decreases. If your systems aren't continuously available, your customers have access to competitive goods and services with just a few clicks, and they may not come back. Global business partners demand access to applications and data around the clock, or they may also choose to do business with your competitors. Meanwhile, strict regulations and service level agreements have been levied against businesses, threatening them with penalties for downtime or lost data. Today's Information Technology professionals face extreme challenges to meet ever-increasing goals for systems availability and business resilience. If these forces are at work in your organization, this paper gives you the information you need to plan for meeting extreme availability challenges head on.

Any discussion of high availability or disaster recovery centers around two key metrics, Recovery Time Objective (RTO) and Recovery Point Objective (RPO). Recovery Time Objective measures the amount of time systems can be down, from the initial outage to the point where business applications are available to users again. Recovery Point Objective (RPO) identifies how much data a business is able and willing to lose or recreate after an outage, measured in time. These are the top service level objectives to consider when designing any high availability or disaster recovery solution.

Organizations must routinely evaluate their RPO and RPO service level agreements (SLAs). Tolerance for data loss in most industries has gone towards zero. Where minutes of downtime were once accepted, the tolerable level of downtime today is also moving towards single-digit minutes or even seconds. Hardware mirroring solutions for IBM i offload the replication function to the storage subsystem and deliver unpredictable recovery times, ranging from minutes to many hours. Traditional software replication solutions for IBM i have delivered the most reliable RTOs in the industry, allowing recovery from unplanned outages in a matter of minutes.

This white paper discusses a paradigm shift in software replication for IBM i that meets Recovery Time Objectives measured in seconds. An overview of the Active-Active replication capabilities in MIMIX Availability is also provided.

Active-Active Replication: A Paradigm Shift

What if the backup server could serve as an additional production server while providing redundancy?

High Availability and Disaster Recovery solutions in use today typically require businesses to recover data or restore operations using a backup copy of the production server that is offline or idle. That backup server contains a copy of the production server's applications, data and system values. Recovering data or restoring operations using the backup server takes time, and the server is often underutilized as it waits to be used for recovery.

What if the backup server could serve as an additional production server while providing redundancy? There would be no "recovery" required and workloads would be more balanced! You simply restore service for the users that were connected to a downed system by pointing them to another server that is up, running and current with all of the updates made on the failed server. With the switch of an IP address, the user is off and running again, resulting in an RTO that is measured in seconds.

This is referred to as an Active-Active replication topology as both the production server and the backup (or target) server are actively serving the business, while also providing redundancy.

Active-Active vs. Active-Passive

Traditional high availability solutions rely on an Active-Passive replication configuration. This configuration is comprised of two servers, a production server (or source server), and a backup server (or target server). The source server is the "active" server, and the target server is the "passive" server. Users are actively running a business application on the source server. The target server, on the other hand, maintains a copy of the business application, but users are not operating on that copy or updating the production database (see Figure 1). Most target servers in an Active-Passive configuration are fully dedicated to HA/DR and are otherwise off-line or idle. Thus the term Active-Passive.

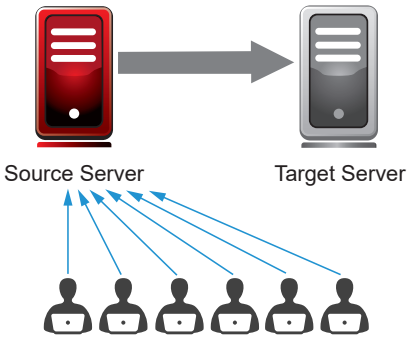


Figure 1: Active-Passive Configuration

Active-Passive solutions require some level of restart on the target server. In the case of hardware replication solutions, the equivalent of a full server restart may be required, depending on the type of solution and nature of the failure. Recovery could take minutes to hours. (Unique considerations for recovery using hardware replication for IBM i are explored in the Vision Solutions white paper, "High Availability for IBM i: Technology and Topology Choices for Ensuring Resilience and Data Integrity.") Software replication solutions eliminate the restart requirement because the target server is a live standby server, but the application must still be restarted. RTO is measured in minutes, which is less than hardware solutions, but more than some industries can tolerate today.

Active-Active solutions eliminate the downtime associated with having to fully recover the target server and restart applications.

The term Active-Active refers to replication configurations where both servers are active production servers, capable of running the same application, and reading and writing the same customer data concurrently (see Figure 2). Both servers participate in replication for HA/DR and also actively function as production servers.

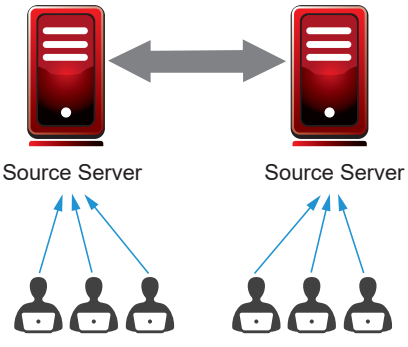


Figure 2: Active-Active Configuration

Active-Active solutions eliminate the downtime associated with having to fully recover the target server and restart applications. As the name implies, the second server is already up and running, the application is active, and the databases are constantly synchronized in real time. And, as discussed later in this paper, there isn't a limit of just two servers. It is possible to have multiple servers that are all active and synchronized.

As the application will be active on two or more servers, and each will be updating the database, the method of database access is a key consideration when changing from a traditional Active-Passive replication solution for IBM i to an Active-Active configuration. The following sections discuss the switch from accessing the database using Positional Replication to Keyed Replication.

Positional Replication

In traditional Active-Passive replication, the standard method of maintaining a copy of the database on the target server is called "Positional Replication." Positional Replication uses row numbers to retrieve and write records (rows). Row numbering is sequential and the position of a record in the table is determined by its row number. A record in the database copy on the target server will have the same record number as the corresponding record in the source database. Updating a record on the target is performed in a single operation and is the most efficient update method.

	SSN	First Name	Last Name	Salary
1	951-64-2nn4	George	Smith	$40,000
2	2NN-79-013N	Ann	Reimer	$80,000
3	472-6n-n971	Mary	Caims	$50,000
4	1n4-0n-17n9	Dennis	Larson	$65,000

Figure 3: Positional Replication is based on row number.

Figure 3 shows a database file with four fields containing social security number, first name, last name, and salary. Row 2 is highlighted for updating. Any read or write operation on this record, whether in the source server database or target server database, is based on this row number. Figure 4 shows an update of the salary field in row 2 on the source server and the corresponding update in row 2 of the database file on the target server.

Source Database

	SSN	First Name	Last Name	Salary
1	951-64-2nn4	George	Smith	$40,000
2	2NN-79-013N	Ann	Reimer	$80,000
3	472-6n-n971	Mary	Caims	$50,000
4	1n4-0n-17n9	Dennis	Larson	$65,000

Update to $85,000

Target Database

	SSN	First Name	Last Name	Salary
1	951-64-2nn4	George	Smith	$40,000
2	2NN-79-013N	Ann	Reimer	$80,000
3	472-6n-n971	Mary	Caims	$50,000
4	1n4-0n-17n9	Dennis	Larson	$65,000

Update to $85,000

Figure 4:
Using Positional Replication record 2 is updated in the source and target database by its row number.

Keyed Replication

In an Active-Active replication environment it is not possible to use Positional Replication. Both servers are active production servers, running the same application and reading and writing the same customer data concurrently. Because records can be inserted into and deleted from the database on both servers at the same time, maintaining identical positions for each record would not be possible.

A new database access method called “Keyed Replication” is required. For Keyed Replication, a unique “key” is used to link the source and target records in the database together (see Figure 5). The unique key is comprised of one or more fields in a table that identify one and only one record in the table. Think of this as a column, or multiple columns, in a database file. The unique key is pre-configured in the database.

In Figure 5, the social security number field is designated as the unique key field. Each record, or row, has a unique value in the key field that identifies that record.

	SSN	First Name	Last Name	Salary
1	951-64-2nn4	George	Smith	$40,000
2	2NN-79-013N	Ann	Reimer	$80,000
3	472-6n-n971	Mary	Caims	$50,000
4	1n4-0n-17n9	Dennis	Larson	$65,000

Figure 5:
Keyed Replication is based on a unique key field.

In addition to delivering an RTO measured in seconds, Keyed Replication also provides the benefit of workload balancing across servers.

Reading and writing a record in the source database still uses the row number, but the corresponding row number in the target database must be determined before a write operation can take place. It is a three-step process to take the record from the source server, find the row number corresponding to the unique data in the key field (the social security number in our example), and write the changed field to the target database using that row number.

Comparing Recovery Time Objectives

Solutions based on Positional Replication can achieve Recovery Time Objectives (RTOs) in the range of minutes. Those solutions rely on a hot backup server, which is fully capable of quickly restarting the business application and resuming transactions. Because changes have been replicated to the target database in real time, it is an identical copy of the source database. Switching to the target server involves completing the application of replicated changes on the target database, switching the direction of the replication, starting the application on the new production server, and redirecting users to the new server. Solutions such as MIMIX Availability automate the switch process to deliver the fastest possible switch times for this method.

In contrast, solutions based on Keyed Replication and running in an Active-Active configuration can achieve an RTO that is faster than that delivered by Positional Replication. Because the application runs on both servers, switching is accomplished by simply redirecting users to the network address of the new server. The result is a switch time that can be measured in seconds.

Workload Balancing

In addition to delivering an RTO measured in seconds, Keyed Replication also provides the benefit of workload balancing across servers. In an Active-Passive environment with Positional Replication, the target server may be an underutilized resource. It maintains a copy of the business application, but users are not operating on that copy or updating the database.

Some environments may lend themselves to having two servers running separate applications and acting as backup servers for one another. However, that scenario does not apply to all environments, and there the target functions only as an HA/DR resource (although some solutions, such as MIMIX Availability, allow access to its data for operations such as queries, reports, backup and testing).

In an Active-Active environment with Keyed Replication, the target is by definition active and not an underutilized resource. The production application workload is shared between the two active servers. This allows servers to be more fully utilized and workloads to be balanced.

Considerations for Active-Active Environments

Achieving increased availability and workload balancing are very attractive reasons for pursuing an Active-Active environment for continuous availability. As you plan an Active-Active implementation, certain considerations must be addressed. Some of these considerations require that your high availability solution provide mechanisms to support the implementation.

MIMIX Availability from Visions Solutions has supported Keyed Replication for Active-Active environments since the 1990s, allowing its customers to achieve unsurpassed high availability. The features of MIMIX that support implementation of an Active-Active configuration are included in the discussion of considerations below.

Collisions

Allowing two active servers to read and write the same application data creates the possibility that two operations could affect the same record at the same time. While this may seem unlikely given the speed of computing, the volume of transactions per second makes it possible. For example, two processes may update a record from one server and delete the same record at the same time from another server. These overlapping or competing operations are called “collisions.”

At a high level, there are a handful of steps that should be taken when implementing an Active-Active environment in order to address the potential for collisions. These steps are outlined below and will be covered in more detail in the following sections.

1. Optimize the application to minimize the possibility of transactions operating on the same record at the same time. An effort should be made to isolate the transactions generated on each server so that they operate on separate sets of records.

   You might also assign users to servers in such a way that the possibility of collisions is minimized. For example, you might handle all calls from clients in one geographical area on one server, while handling calls from clients in another geography on a second server. This can drastically reduce the window of opportunity for collisions to occur.

2. In an ideal world, applications would be optimized to completely avoid collisions. However, in reality there will be a small window of opportunity. Therefore, you must analyze the remaining collision points as they relate to your application. It is possible that collisions could be detected during database insertions, changes or deletions. What types of collisions might occur with your application? Don’t forget to factor in human error, such as typing the wrong number in a key field.

3. Define a systematic process for resolving collisions. It is critical that Active-Active replication solutions provide collision detection and resolution mechanisms to support this effort. MIMIX Availability provides robust, proven collision detection and resolution capabilities to aid in implementing your process, including methods for handling simultaneous deletes or non-overlapping field changes.

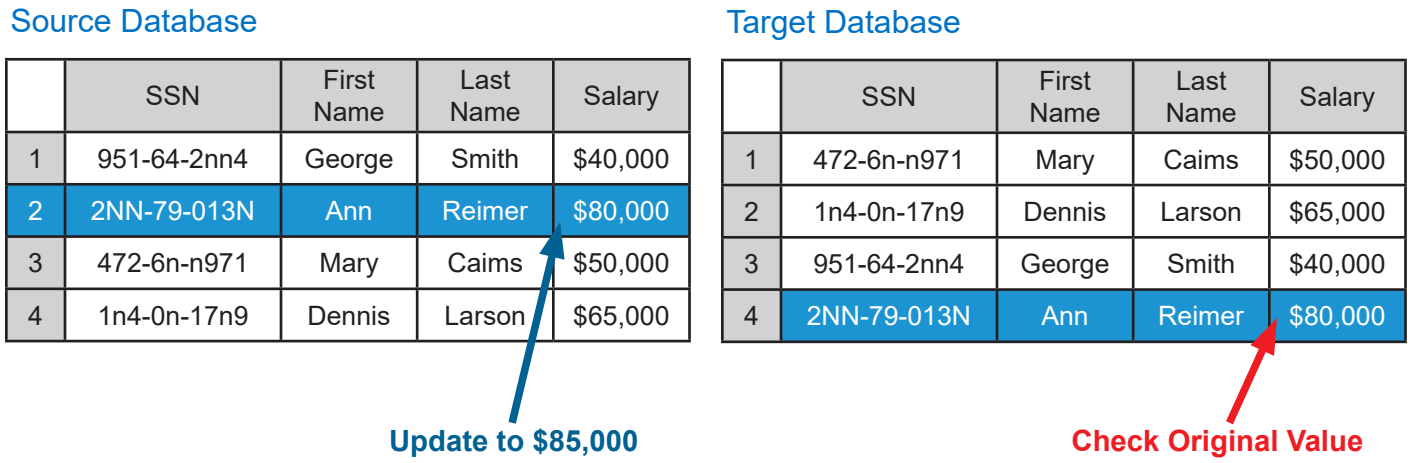
4. Identify situations unique to your application that require special rules for handling. The set of rules provided by MIMIX can be extended with these custom rules.

It is important that your high availability solution provide methods to resolve conflicts, and that your vendor work closely with you during this implementation.

Collision Detection

MIMIX Availability uses the key field capabilities of the IBM i OS database to enable Keyed Replication, and uses information contained in the journal entry to detect and resolve collisions.

Figure 6 shows an example transaction which updates a field in a record on the source server. With Keyed Replication, the record with the matching social security number in the database on the target server (shown on the right) must be checked to make sure it has not changed since the transaction started on the source server.



	SSN	First Name	Last Name	Salary
1	951-64-2nn4	George	Smith	$40,000
2	2NN-79-013N	Ann	Reimer	$80,000
3	472-6n-n971	Mary	Caims	$50,000
4	1n4-0n-17n9	Dennis	Larson	$65,000

	SSN	First Name	Last Name	Salary
1	472-6n-n971	Mary	Caims	$50,000
2	1n4-0n-17n9	Dennis	Larson	$65,000
3	951-64-2nn4	George	Smith	$40,000
4	2NN-79-013N	Ann	Reimer	$80,000

Figure 6:
To detect collisions using Keyed Replication, the target record's data must be checked before an update.

To make this comparison, MIMIX must know the original value of the field being changed when it is applying that change to the target database. This is accomplished using the functionality found in journaling.

To detect collisions, MIMIX uses the “before image” of the record and the “after image” of the record. These images are available in the journal that is sent to the target server at the completion of each transaction (see Figure 7). The before image is used to find the unique key field for the record being changed and the original value for all of the fields in the record. The entire record from the source server in the before image should be identical to the record retrieved from the target database. In the case of Figure 7, the salary field in the record with the same social security number key does not match. That means that the value has changed on the target since the transaction started on the source, and we have detected a collision.

Source Database

	SSN	First Name	Last Name	Salary
1	951-64-2nn4	George	Smith	$40,000
2	2NN-79-013N	Ann	Reimer	$80,000
3	472-6n-n971	Mary	Caims	$50,000
4	1n4-0n-17n9	Dennis	Larson	$65,000

Update to $85,000

Before Image

2	2NN-79-013N	Ann	Reimer	$80,000

After Image

2	2NN-79-013N	Ann	Reimer	$85,000

Target Database

	SSN	First Name	Last Name	Salary
1	472-6n-n971	Mary	Caims	$50,000
2	1n4-0n-17n9	Dennis	Larson	$65,000
3	951-64-2nn4	George	Smith	$40,000
4	2NN-79-013N	Ann	Reimer	$82,000

Target Value Does Not Match Before Image. Collision!

Figure 7:
Collisions are detected with Keyed Replication by comparing the journal’s before image for a record with the fields in the target record. If a target field does not match, there has been a collision.

It is possible for collisions to occur in three replication scenarios: during insertions, changes or deletions. It is in these situations where collision detection mechanisms are active.

- **Insertions:** When inserting a new record in the source database with a new key value for that database, the same insertion may be happening at the target database. When replication inserts the new record on the target server, it must check to see that the record does not already exist.

- **Updates:** When a record in the source database is updated, collisions can be detected at three points when the change is replicated to the target database – when finding the record, comparing the fields in the record, and updating the record.

- **Deletions:** When a record is deleted from the source database, collisions can be detected at three points while attempting to replicate the deletion to the target database – when finding the record to delete, comparing the record with the expected original value, and when deleting the record.

Collision Resolution

As mentioned above, it is critical that your high availability solution provide collision resolution mechanisms to support an Active-Active implementation. MIMIX Availability provides a set of actions for resolving the standard collisions that might occur when multiple production servers are operating on a single record at the same time. These methods include:

- Ignoring an update if the target database has already been updated to the same value by another production server.

- Ignoring a delete if the record has already been deleted on the target database by another production server.

- Merging changes if the changes are for non-overlapping fields within the database record.

- Calling a custom exit program to resolve collisions that are unique to the application.

An example process for defining a collision resolution strategy might be:

1. Identify all collision points. As mentioned above, collisions points are generally found where records are inserted, updated or deleted.

2. Determine all the possible failure events that could occur at each collision point. For example, Figure 6 showed a collision point where simultaneous updates could be performed to the same field of a record from both servers.

3. Outline the order of the methods that will be used to attempt resolution. The four MIMIX collision resolution methods described previously are commonly used in that order.

4. Define any custom resolution methods required for your application. MIMIX provides an exit point for calling custom methods. You might program the method yourself or work with your vendor.

5. Finally, determine what should happen when all attempted resolution methods fail. For example, you might put the file on hold until the conflict can be resolved by an operator.

Multi-Node Environments

MIMIX Availability in an Active-Active configuration is fully compatible with MIMIX Global, the MIMIX add-on that enables multiple servers in a high availability environment to be switched as easily as a two-node environment.

When using MIMIX Global in an Active-Passive implementation, a single source server can replicate to target servers intended for high availability, remote disaster recovery, or to serve as real-time copies of the database for queries, reports or other applications. In an Active-Active environment, MIMIX Global supports multiple source servers in the same manner. Figure 8 shows a sample of a four node Active-Active topology.

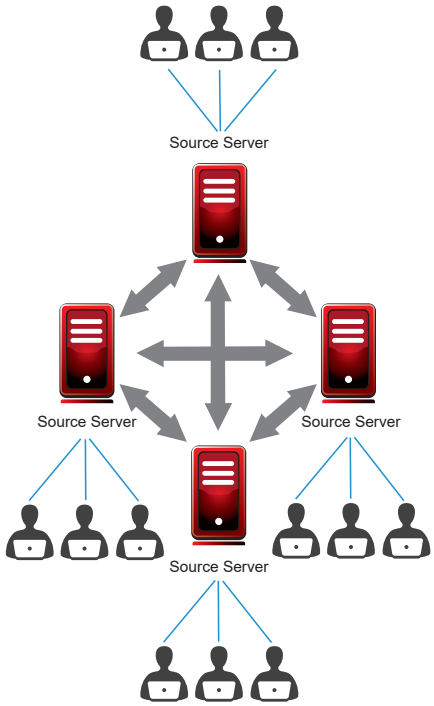


Figure 8:
MIMIX Availability supports three or more source servers in an Active-Active environment when paired with MIMIX Global.

MIMIX Availability Is the Clear Choice

Keeping up with expectations for systems availability and business resilience imposed by customers, partners and external regulatory bodies requires a paradigm shift. For organizations whose Recovery Time Objective is now measured in single-digit minutes or even seconds, moving from a high availability solution based on traditional Positional Replication to an Active-Active implementation using Keyed Replication enables them to meet extreme availability objectives and balance server workloads.

To achieve increased availability and workload balance, certain conditions must be addressed, primarily related to minimizing and resolving the collisions that can occur when multiple active servers operate on the same database. It is crucial to work with a high availability solution provider, such as Vision Solutions, who can guide you through an Active-Active implementation. MIMIX Availability, Vision's market-leading high availability solution for IBM i, has provided support for Active-Active configurations using Keyed Replication since the 1990s. Expertise developed over many years along with the proven, reliable collision detection and resolution methods built into the product make MIMIX Availability the clear choice for achieving unsurpassed high availability objectives.

Easy. Affordable. Innovative. Vision Solutions.

Vision Solutions is a leading provider of business resilience solutions—high availability, disaster recovery, migration and data sharing—for IBM Power Systems. For more than 25 years, customers and partners have trusted Vision to protect and modernize their IT environments, whether on-premises or in the cloud.

Visit visionsolutions.com and follow us on social media, including Twitter, Facebook and LinkedIn.

Find us on:

Facebook:	http://www.facebook.com/VisionSolutionsInc
Twitter:	http://twitter.com/VSI_Power
YouTube:	http://www.youtube.com/VisionSolutionsInc
Vision Solutions Blog:	http://www.visionsolutions.com/blog

Midland Information Systems, Inc

VISION® SOLUTIONS

MIMIX® AVAILABILITY™

15300 Barranca Parkway
Irvine, CA 92618
1.949.253.6500
1.800.683.4667

visionsolutions.com