HIGH AVAILABILITY STRATEGIES

HA Strategies for InterSystems Caché, Ensemble, and HealthShare Foundation

Introduction........................................................................................................................................... 1
Operating System Failover Clustering ............................................................................................... 2
Virtualization-Based High Availability.............................................................................................. 3
Caché Database Mirroring .................................................................................................................. 4
  Mirroring Failover Strategies ........................................................................................................... 5
  Failover with ISCAgent Only (Default Configuration) ................................................................. 5
  Failover with Custom Solution: Reliable Network Ping............................................................. 6
  Failover with Custom Solution: Other Strategies ........................................................................... 7
Hybrid HA Strategy ............................................................................................................................ 8
General System Outages ................................................................................................................... 9
  Planned Outage Types .................................................................................................................. 10
  Unplanned Outage Types ............................................................................................................. 10
Appendix A: Sample Reliable Network Configurations ................................................................... 12
Appendix B: Hybrid HA Solution ...................................................................................................... 14
  Virtualization & Mirroring Hybrid .............................................................................................. 14
  OS Failover Clustering & Mirroring HYBRID ............................................................................... 17
Appendix C: Sample ^ZMIRROR for Reliable Network Ping Failover ............................................. 18
Appendix D: Manual Failover after Unplanned Outage of Primary ................................................ 19

Vik Nagjee, Product Manager (vik.nagjee@intersystems.com)
Ray Fucillo, Solutions Engineer (ray.fucillo@intersystems.com)
Mark Bolinsky, Technology Architect (mark.bolinsky@intersystems.com)

July 1, 2013
INTRODUCTION

This document is intended to provide a survey of various High Availability (HA) strategies that can be used in conjunction with InterSystems Caché, Ensemble, and HealthShare Foundation. This document also provides an overview of the various types of system outages that can occur, as well as how each strategy would handle a given outage, with the goal of helping you choose the right strategy for your specific deployment.

The strategies surveyed in this document are based on three different HA technologies: Operating System Failover Clusters, Virtualization-Based HA, and Caché Database Mirroring. Table 1 below highlights some key differences between these technologies.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Caché Database Mirroring</th>
<th>Operating System Failover Clustering</th>
<th>Virtualization High Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failover after Machine Power Loss or Crash</td>
<td>By default, no automatic failover if primary machine is unreachable (crash, power loss, network loss); anything else requires careful planning and configuration</td>
<td>Handles machine failure seamlessly</td>
<td>Handles physical and virtual machine failures seamlessly</td>
</tr>
<tr>
<td>Protection from Storage Failure and Corruption</td>
<td>Built-in replication protects against storage failure; logical replication avoids carrying forward many types of corruption</td>
<td>Relies on shared storage device, so failure is disastrous; storage-level redundancy optional, but can carry forward some types of corruption</td>
<td>Relies on shared storage device, so failure is disastrous; storage-level redundancy optional, but can carry forward some types of corruption</td>
</tr>
<tr>
<td>Failover after Caché Shutdown, Hang, or Crash</td>
<td>Rapid detection and failover if Caché is shut down or hung</td>
<td>Can be configured to fail over after Caché outage</td>
<td>Can be configured to fail over after Caché outage</td>
</tr>
<tr>
<td>Caché Upgrades</td>
<td>Allows for minimum-downtime Caché upgrades*</td>
<td>Caché upgrades require downtime</td>
<td>Caché upgrades require downtime</td>
</tr>
<tr>
<td>Application Mean Time to Recovery</td>
<td>Failover time is typically seconds</td>
<td>Failover time can be minutes</td>
<td>Failover time can be minutes</td>
</tr>
<tr>
<td>External File Synchronization</td>
<td>Only databases are automatically replicated; external files need third party solution</td>
<td>All files are available to both nodes</td>
<td>All files available after failover</td>
</tr>
</tbody>
</table>

* Requires a configuration in which application code, routines, and classes are in databases separate from those that contain application data
A very common approach to achieving HA is to use failover solutions that are provided at the operating system level. Examples of such solutions exist on all platforms and include Microsoft Windows Clusters, HP Serviceguard, Veritas Cluster Server, and IBM SystemMirror (PowerHA), as well as the respective clustering packages from Red Hat and SUSE Linux. While the specifics of the configuration may differ slightly among the various platforms, the model is generally the same: two identical servers with a shared storage device (often a SAN or iSCSI targets) and a shared IP address, one actively serving production workload, and one standing by in case of failure. When an outage occurs on the active system, the failover technology transfers control of the shared disk and the shared IP address to the standby node, and then starts application services, including Caché.

Caché is designed to integrate easily with these failover solutions. The production instance of Caché is installed on the shared storage device so that both members of the failover cluster recognize the instance, then added to the failover cluster configuration so that it will be started automatically as part of failover. When Caché starts on the newly active node during failover, it automatically performs the normal startup recovery from WIJ and journal files (again, located on the shared storage device); data integrity is preserved just as though Caché had simply been restarted on the original failed node.

Pros:
- Handles machine failure seamlessly
- Most common HA choice
- Available on all supported platforms through OS or 3rd party vendors
- All files (database and external) available to both nodes

Cons:
- Storage failure is disastrous
- Upgrades require downtime
- Application Mean Time to Recovery can be minutes

The appendixes in the Caché High Availability Guide contain detailed information on how to correctly configure Caché with some of the more popular OS failover clusters.
VIRTUALIZATION-BASED HIGH AVAILABILITY

Virtualization technologies, such as VMware vSphere ESX/ESXi, provide High Availability capabilities, which typically monitor the overall health and viability of the physical hardware, as well as the guest operating systems running therein. On failure, the Virtualization HA software will automatically restart the failed virtual machine on an alternate surviving hardware. When Caché restarts, it automatically performs the normal startup recovery from WIJ and journal files; data integrity is preserved just as though Caché had simply been restarted on the original failed node.

In addition, guest operating systems can be relocated to other servers within the virtual environment, allowing for a virtual machine to be uplifted to alternate physical infrastructure, for maintenance purposes, without downtime. This feature is available as VMware vMotion, IBM Live Partition Mobility, HP Live VM Migration, and others.

Pros:  
- Handles machine failure seamlessly  
- Most common HA choice in virtual environments  
- All files are available after failover  
- Planned physical hardware maintenance requires little or no application downtime

Cons:  
- Storage failure is disastrous  
- Software upgrades require downtime  
- Application Mean Time to Recovery can be minutes for unplanned hardware failures

Proper infrastructure is required to effectively support high availability in a virtual environment. This includes storage, networking, and processor capacity. Please refer to your virtualization supplier’s documentation for best practices.
A mirror consists of two physically independent Caché systems, called *failover members*. The mirror automatically assigns the role of *primary* to one of the failover members, while the other member automatically becomes the *backup* system. Data is replicated from the primary to the backup failover member, thus providing built-in data redundancy. Caché Database Mirroring (Mirroring) is designed to provide an economical solution for rapid, reliable, robust, automatic failover between two Caché systems for planned and unplanned outages.

Mirroring additionally allows asynchronous replication to other members called *async members*. Async members can be used to meet a variety of demands including disaster recovery, reporting, data warehousing, and business intelligence. Async members are not available for automatic failover, but async members that are designated for disaster recovery can be quickly promoted to take over as part of your disaster recovery procedures. For more information on the disaster recovery features of mirroring (specific to versions 2013.1 and later), see the Caché documentation section on [Promoting a DR Async Member to Failover Member](#) and [Mirror Outage Procedures](#). The remainder of the discussion of mirroring in this document pertains to failover members and the high availability features of mirroring.

**Pros:**
- Automatic failover if Caché is shut down or hung on the primary
- Allows for minimum-downtime Caché upgrades (zero-downtime may be achievable with a specific configuration)
- Data replication protects against storage failure on the primary
- Failover time is typically seconds, providing fast application mean time to recovery
- Can be less expensive than clustering solutions
- Logical data replication can protect against physical corruption being carried forward to the other system
- Async members for disaster recovery and reporting allow you to meet multiple needs with one technology.

**Cons:**
- Default configuration does not automatically failover if the primary machine is unreachable (due to OS crash, power loss, network outage, and so on)
- Always-automatic failover requires careful planning and configuration
- Only databases are automatically replicated; external files needed by the application (i.e., file streams, images, etc.) need a third party replication solution
- Security and configuration management is currently decentralized
General High Availability Strategies: InterSystems Caché, Ensemble, and HealthShare Foundation

MIRRORING FAILOVER STRATEGIES

Mirroring can be used to meet a variety of high availability needs. The strategy for meeting these demands will encompass the mirroring settings, hardware configuration, data center configuration, manual procedures, and sometimes a custom ^ZMIRROR routine.

In all cases, in order to take over as primary, it must be definitively determined, automatically through software or through manual intervention, that the primary failover member is down, and that the backup failover member has all of the journal data that the primary failover member has durably committed. Mirroring includes an executable program, ISCAgent, which runs on all mirror members and is used in making this determination. Mirroring also allows you to optionally create an alternate mechanism to determine whether the primary failover member is down.

The remainder of this section describes the various mirroring failover strategies. The general mirroring pros and cons listed above apply to each of the failover strategies; specific pros and cons for each strategy are separately listed below.

FAILOVER WITH ISCAGENT ONLY (DEFAULT CONFIGURATION)

When the backup mirror member detects a failure of the primary, it attempts to contact the ISCAgent on the primary machine. If the backup successfully contacts the ISCAgent, it can then confirm that the primary is down or force it down if it is unresponsive, download any journal information required for it to be fully caught up, and safely take over as primary.

If the ISCAgent cannot be contacted (for example, if the primary server is down), failover does not occur. Of course, the administrator can take manual steps to confirm that the primary is down and that the backup has the necessary journal data, then initiate failover. See Appendix D for instructions on Manual Failover After Unplanned Outage of Primary.

Pros:

- Easiest mirroring failover strategy to implement (this is the default configuration)
- Completely safe failover; no risk of split-brain (that is, two servers both acting as primary)
- No specialized hardware or software needed
- Allows rapid failover from Caché shutdown, an unresponsive Caché instance, and many hardware or software failures that prevent Caché from working, so long as the ISCAgent remains reachable from the backup member
- Failover members may be in separate data centers, possibly allowing for HA and DR goals to be met with only two servers (allowable latency is dependent on the application)

Cons:

- No automatic failover occurs after failure that renders the ISCAgent unreachable, such as host failure
- If the primary host is unavailable, it can be difficult to determine whether the backup has all the required journal information in order to verify that it is safe to initiate manual failover

To implement this strategy, leave the Agent Contact Required for Takeover configuration setting at YES (the default).


General High Availability Strategies: InterSystems Caché, Ensemble, and HealthShare Foundation

FAILOVER WITH CUSTOM SOLUTION: RELIABLE NETWORK PING

When the backup mirror member detects a failure of the primary, it attempts to contact the ISCAgent on the primary machine. If the backup successfully contacts the ISCAgent, it can then confirm that the primary is down or force it down if it is unresponsive, download any journal information required for it to be fully caught up, and safely take over as primary.

If the ISCAgent cannot be contacted (for example, if the primary server is down), network pings over the public and private network are utilized to determine the status of the primary server (this requires custom programming which is implemented in $\$IsOtherNodeDown^ZMIRROR()). If the primary does not respond to the pings on either the public or the private network, the backup assumes that the primary is down, and takes over as primary. Because the lack of ping response from the primary server does not strictly guarantee that the server is down, there is a risk of split-brain (two servers simultaneously acting as primary) that cannot be completely eliminated with this strategy. Other mirroring failover strategies discussed in this document carry no risk of split-brain. To minimize the risk, the following is required:

- The networking between the failover members must be redundant, reliable, and highly available.
- The failover members should be hosted directly on physical machines, not on a virtualization platform; on virtualized platforms, activity at the host/hypervisor level may cause a member to become temporarily unresponsive to ping while it is still running. See the Hybrid HA Strategy for information on how to safely extend mirroring in its default configuration to provide higher availability in a virtualized environment.

Pros:

- Allows rapid failover following server/host failure

Cons:

- Requires implementation of a custom ^ZMIRROR routine (InterSystems can provide a sample).
- Requires specialized networking hardware configuration to provide very robust networking.
- Recommended that the failover machines are located in the same data center to avoid network isolation.
- The risk of split-brain (two servers both acting as primary) cannot be completely eliminated.

To implement this strategy:

1. Create a hardware configuration that provides an extremely reliable network between the primary and backup failover members. Please reference Appendix A for an example of a reliable network configuration between two failover members.
2. Customize the sample implementation of $\$IsOtherNodeDown^ZMIRROR() from the routine provided in Appendix C. In any scenario under which the ping mechanism cannot adequately determine that the primary is down, your implementation must assume that it is up so that automatic failover will not occur. Of course, the administrator can take manual steps to determine that the primary is down and that the backup has the necessary journal data, and then initiate failover. See Appendix D for instructions on Manual Failover After Unplanned Outage of Primary
3. Set Agent Contact Required for Takeover to NO.
4. Adjust Trouble Timeout Limit to allow sufficient time for the $\$IsOtherNodeDown^ZMIRROR mechanism to operate. For example, while testing failover you may notice a message similar to the following in the cconsole.log file on the backup failover member: Mirror recovery time of 7.101 seconds exceeded trouble timeout of 6 seconds. Restarting. In this example, you might consider increasing the Trouble Timeout Limit to 8 seconds.


FAILOVER WITH CUSTOM SOLUTION: OTHER STRATEGIES

When the backup mirror member detects a failure of the primary, it attempts to contact the ISCAgent on the primary machine. If the backup successfully contacts the ISCAgent, it can then confirm that the primary is down or force it down if it is unresponsive, download any journal information required for it to be fully caught up, and safely take over as primary.

If the ISCAgent cannot be contacted (for example, if the primary server is down), an alternate procedure runs to determine whether the primary is down using a mechanism customized for the deployment.

Pros:

↑ Allows a custom-tailored solution to match your environment

↓ Requires engineering a custom mechanism for determining that the primary is down when the ISCAgent cannot be contacted, and programming $$IsOtherNodeDown^ZMIRROR()$$ to use that mechanism

↓ May require specialized hardware

Cons:

To implement this strategy:

1. Create a configuration that provides for a robust mechanism to definitively determine that the primary is down. The architecture used to create such a configuration will likely depend on the location and proximity of the two failover members.

2. Program $$IsOtherNodeDown^ZMIRROR()$$ to use the custom mechanism to determine whether the primary is down. If the chosen mechanism cannot definitely determine that the primary is down, it must be assumed that it is up so that automatic failover will not occur. Of course, the administrator can take manual steps to determine that the primary is down and that the backup has the necessary journal data, and then initiate failover. See Appendix D for instructions on Manual Failover After Unplanned Outage of Primary.

3. Set Agent Contact Required for Takeover to NO.

4. Adjust Trouble Timeout Limit to allow sufficient time for the $$IsOtherNodeDown^ZMIRROR()$$ mechanism to operate.
HYBRID HA STRATEGY

Database Mirroring can be used in conjunction with either failover clustering or virtualization HA to provide extremely robust high availability strategies for planned and unplanned outages.

Failover clustering or virtualization HA provides automatic failover for machine or OS-level failures (unplanned outages). Database Mirroring (using its default configuration) provides automatic failover for all types of planned outages as well as unplanned Caché outages; its built-in data replication protects against storage failures.

Pros:  

- Provides safe automatic failover for planned and unplanned outages of hardware, OS, and the database  
- Available on all supported platforms through OS or third party vendors  
- Planned physical hardware maintenance requires little or no application downtime  
- Mirroring replication protects against storage failures  

Cons:  

- High complexity  
- Potentially costly to implement  
- Multiple technologies to manage and learn  

To implement this strategy, see the Hybrid HA Solution section in Appendix A for some configuration options. Note that you should leave the Agent Contact Required for Takeover Mirroring configuration as YES (the default).
There are two broad categories of system outages: planned and unplanned. Which HA strategy you choose will largely determine the likelihood of automatic failover during a given type of outage. Table 1 summarizes the various types of outages and indicates whether each HA strategy would likely fail over automatically in response to that particular outage.

<table>
<thead>
<tr>
<th>Planned Outage</th>
<th>Caché Upgrade</th>
<th>OS Upgrade</th>
<th>Application Upgrade</th>
<th>Server Upgrade/Maintenance</th>
<th>Storage Upgrade/Maintenance</th>
<th>Caché hang, crash, or shutdown</th>
<th>All networking fails on primary</th>
<th>Public network interface fails on primary</th>
<th>Server (host) failure, crash, or power loss</th>
<th>Storage failure</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS Failover Cluster</td>
<td>NO</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>YES</td>
<td>NO</td>
<td>MAYBE</td>
<td>MAYBE</td>
</tr>
<tr>
<td>Virtualization HA</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>YES</td>
<td>NO</td>
<td>MAYBE</td>
<td>MAYBE</td>
</tr>
<tr>
<td>Mirroring: Failover w/ ISCAgent Only</td>
<td>YES</td>
<td>YES</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>NO</td>
<td>N/A†</td>
<td>MAYBE</td>
<td>YES</td>
</tr>
<tr>
<td>Mirroring: Failover w/ Custom Solution - Reliable Network Ping</td>
<td>YES</td>
<td>YES</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>YES</td>
</tr>
<tr>
<td>Mirroring: Failover w/ Custom Solution - Other Strategies</td>
<td>YES</td>
<td>YES</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
<td>MAYBE</td>
</tr>
<tr>
<td>Hybrid Strategy</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
</tbody>
</table>

Table 2: Automatic Failover by Outage Type and HA Strategy

The following sections describe the reasoning for the outcomes in Table 2.

† This is a special case that can result in split-brain (two servers both acting as primary). See additional details below.
**PLANNED OUTAGE TYPES**

- **Caché Upgrade**: OS Failover Clusters and Virtualization HA require downtime for Caché upgrades because there is only one instance of Caché in this configuration. Mirroring, on the other hand, allows you to perform a rolling upgrade between compatible versions by upgrading Caché on the backup failover member first, then failing over, and upgrading the other node. Since the Hybrid Strategy includes Mirroring, the same benefit applies to this HA strategy.
- **OS Upgrade**: All outlined HA strategies, except Virtualization HA, typically provide rolling upgrade options; Virtualization HA requires downtime because there is only one instance of the guest OS.
- **Application Upgrade**: OS Failover Clusters and Virtualization HA require downtime for application upgrades because there is only one production instance of Caché in this configuration. Mirroring may provide a way to minimize the downtime for application upgrades, assuming no data conversion is necessary (before, after, or during the upgrade), and assuming that the application code and data are stored in separate databases (a best practice). Since the Hybrid Strategy includes Mirroring, the same benefit applies to this HA strategy.
- **Server Upgrade/Maintenance**: All outlined HA strategies typically provide rolling maintenance and upgrade options.
- **Storage Upgrade/Maintenance**: OS Failover Clusters and Virtualization HA may require downtime for storage upgrades or maintenance because there is one shared storage device in this configuration. Mirroring, on the other hand, typically uses independent storage devices, allowing you to perform rolling storage upgrades or maintenance. Since the Hybrid Strategy includes Mirroring, the same benefit applies to this HA strategy.

**UNPLANNED OUTAGE TYPES**

- **Caché hang, crash, or shutdown**: OS Failover Clusters and Virtualization HA provide monitoring scripts that can be configured to induce failover in the event of Caché failure. Mirroring immediately detects this condition and triggers an automatic failover. Since the Hybrid Strategy includes Mirroring, the same benefit applies to this HA strategy.
- **All networking fails on primary**: OS Failover Clusters and Virtualization HA typically detect network loss and automatically trigger a failover. Mirroring behaves differently based on the strategy that has been deployed:
  - **Failover with ISCAGENT Only**: Since contact with the ISCAGENT on the failed node is not possible, mirroring does not automatically fail over.
  - **Failover with Reliable Network Ping**: This strategy is predicated on having a 100% reliable/redundant network, so failures of this type should be precluded. If a failure of this type were to occur, the $ISOtherNodeDown^ZMIRROR() procedure would allow mirroring to automatically fail over, resulting in split-brain (two servers both acting as primary).
  - **Failover with Custom Solution**: The success of automatic failover depends on the custom implementation of $ISOtherNodeDown^ZMIRROR().
  - **Hybrid Strategy**: Since the Hybrid Strategy includes OS Failover Clusters / Virtualization-based HA, automatic failover occurs with this sort of failure.
- **Public network interface fails on primary**: OS Failover Clusters and Virtualization HA typically detect network loss and automatically trigger a failover. Mirroring, by default, does not detect public network interface failures. However, it is possible for you to create a custom monitoring script to induce failover in the event of public network interface failure on the primary server; while this would apply to any of the mirroring strategies, it requires that a separate (private) network be configured between the two failover members. Since the Hybrid Strategy includes OS Failover Clusters / Virtualization-based HA, automatic failover occurs with this sort of failure.
- **Server (host) failure, crash, or power loss**: OS Failover Clusters and Virtualization HA detect these types of failures and automatically fail over. Mirroring behaves differently based on the strategy that has been deployed:
  - **Failover with ISCAGENT Only**: Since contact with the ISCAGENT on the failed node is not possible, mirroring does not automatically fail over.

---

‡ Some SAN storage systems allow for non-disruptive updates (NDU), which may allow storage upgrade/maintenance without downtime.
- **Failover with Reliable Network Ping**: The $\$IsOtherNodeDown^ZMIRROR()$ procedure will detect that the primary is down, allowing mirroring to automatically fail over.
- **Failover with Custom Solution**: The success of automatic failover depends on the custom implementation of $\$IsOtherNodeDown^ZMIRROR()$.
- **Hybrid Strategy**: Since the Hybrid Strategy includes OS Failover Clusters / Virtualization-based HA, automatic failover occurs with this sort of failure.

**Storage failure**: OS Failover Clusters may be able to tolerate a storage failure if disk-based replication is utilized; otherwise, a storage failure is catastrophic, as both nodes would be operating on a shared storage device. Similarly, Virtualization HA may be deployed on replicated storage. Mirroring, assuming the failover members are configured to use independent storage devices, may be able to automatically fail over in the event of storage failure. The ability to automatically fail over may depend on the nature of the storage failure. The data replication provided by Mirroring protects production data from storage failures, allowing you to manually induce failover if needed. Since the Hybrid Strategy includes Mirroring, the same benefit applies to this HA strategy.
APPENDIX A: SAMPLE RELIABLE NETWORK CONFIGURATIONS

This appendix describes various possible reliable network configurations used to deploy Mirroring.

The **Mirroring Failover with Custom Solution: Reliable Network Ping** strategy requires that the networking between the two failover members is redundant, reliable, and highly available in order to prevent split-brain (i.e., two servers both acting as primary). The following sample configurations are examples.

Figure 1 illustrates a very simple configuration whereby the two failover members are directly connected to each other with crossover cables for mirror communications over a private network, entirely avoiding an external switch or hub; this reduces the likelihood of a network-related failure between the two systems. The systems are also connected to an external bank of switches for public communication. This configuration is highly economical and simple in design.

A few additional notes about the sample configuration presented in Figure 1 follow:

- Each system is shown with two dual-ported NICs. This is for reliability and redundancy purposes, and is a recommended configuration.
- Support for using bonded or teamed interfaces ports with crossover cables is OS and NIC driver dependent. Not all operating systems and NIC drivers support this type of bonding. Please refer to system documentation for compatibility.
Note that the two failover members could either be configured with internal storage (SAS), or have external storage (SAN, NAS, and so on); in the case of a shared SAN, however, it is recommended that the two systems use independent physical spindles.

This example could be extended to include highly available network switching, which will mitigate the physical co-location requirement; however, it is imperative that the networking be reliable. Figure 2 illustrates a sample configuration using dual computer rooms (same campus) each with highly available stacked switching using EtherChannel bonds.
APPENDIX B: HYBRID HA SOLUTION

OS Failover Clusters and Virtualization-based HA technologies provide automatic failover for system outages (such as power loss, OS crash, or network loss); Mirroring provides replication as well as built-in automatic failover for database-related outages, including upgrades, maintenance, or Cache failure. A Hybrid HA solution is one that leverages the strengths of both types of HA technologies – OS Failover Clusters or Virtualization-based HA plus Cache Database Mirroring – to provide very high levels of availability. These configurations, however, can be complex, and require careful planning.

Mirroring should be configured in the default (Agent Contact Required for Takeover = YES) mode so that it will defer to the virtualization HA or the OS failover cluster for hardware/server-level outages (unplanned), while still rapidly failing over to the mirror backup member for database outages (planned or unplanned).

VIRTUALIZATION & MIRRORING HYBRID

The use of virtualization carries with it additional requirements to provide acceptably high availability. When configuring the guest virtual machines, each of the mirror members should have anti-affinity rules defined. This will ensure that the primary and backup failover members will avoid running on the same physical server.

Installing a minimum of six (6) Ethernet interfaces spread across at least two (2) or three (3) physical adapters is highly recommended. In blade server technologies, the use of VICs (Virtual Interface Cards) or CNAs (Converged Network Adapters) is common, and support both Ethernet and Fiber Channel communications, so consult your respective hardware supplier’s best practices for virtual interface definitions.

Figure 3 illustrates a sample 3-node VMware ESX® cluster with redundant networking.

---

§ Please note VMware is used only as an example. Other virtualization technologies from IBM, HP, Microsoft, and Linux KVM variants such as Red Hat RHEV-M that supports automatic partition or virtual machine mobility/restart can be used as well.
Along with redundancy in the network, added attention to storage is highly recommended to maintain isolation for each mirror member. To provide isolation, each mirror member should use separate data-stores which should be at a minimum on separate disk groups in a single SAN storage array or ideally on two separate storage arrays altogether. Figure 4 illustrates the storage layer isolation in a virtualized environment.

![Figure 4: Redundant Storage with VMware vSphere](image-url)
Figure 5 illustrates VMware HA recovering a failed virtual machine from other physical resources. Here, a minimum of three (3) servers is required.

Figure 5: VMware HA with Database Mirroring

**Please note VMware is only used as an example. Other virtualization technologies from IBM, HP, Microsoft, and Linux KVM variants such as Red Hat RHEV-M that supports the automatic partition or virtual machine mobility/restart can be used as well.**
OS FAILOVER CLUSTERING & MIRRORING HYBRID

Figure 6 illustrates failover clustering providing hardware failure protection for each mirror member. Here, a minimum of four (4) physical servers is required, so as to deploy two 2-node failover clusters.

Figure 6: OS Failover Clustering & Mirroring Hybrid
APPENDIX C: SAMPLE ^ZMIRROR FOR RELIABLE NETWORK PING FAILOVER

A sample ^ZMIRROR routine is provided by InterSystems in order to assist you to implement the Mirroring Reliable Network Ping Failover strategy.

The latest sample file can be located at the following URL:

Download this file to your environment, and load it into the %SYS namespace as follows:

Do $System.OBJ.Load("<path>/ZMIRROR_RELIABLE_NETWORK_PING.xml","ck")

Once you have loaded this file, you will need to customize it to fit your environment. The routine itself contains self-explanatory steps and comments.

Please contact the InterSystems Worldwide Response Center for clarification, if needed.
APPENDIX D: MANUAL FAILOVER AFTER UNPLANNED OUTAGE OF PRIMARY

This appendix describes how to manually induce failover after an unplanned outage of the primary. It is used in conjunction with the strategies listed previously in this document when the software configuration alone cannot determine that it is safe to automatically fail over.

Here, we assume that the primary system has completely failed. If the primary system is up, the backup system will automatically contact the ISCAgent on the primary to determine the status and possibly take over. In general, if simply restarting the failed primary is possible, it is preferred over any of the following manual failover steps.

If you have access to the journal files from the failed/down primary, you can use the following procedure without risking data inconsistency between the primary and backup:

1. Ensure that the primary machine is down and remains down during this procedure.
2. Shut Caché down on the backup machine.
3. Copy the necessary journal files to the backup. That is, re-copy the last mirror journal file that the backup has, and any newer mirror journal files that the primary may have created.
4. Delete the Mirror Journal Log file on the backup. This file is located in the mgr (manager) directory of the instance, and is named as follows: mirrorjrn-{MirrorName}.log. This will automatically be rebuilt in a later step.
5. Start up Caché on the backup machine.
6. Use the Force Become Primary function on the backup to become primary. This option is available from the ^MIRROR utility.

If you don’t have access to the journal files from the failed/down primary, there may be a risk of data inconsistency between the primary and backup. If all of the following conditions are known with certainty, then a manual failover is safe to perform because the backup has all of the journal data from the primary as of the time of the failure:

- The primary server is down.
- The time when the backup disconnected from the primary matches the time that the primary server went down. The certainty with which you are able to make this determination is dependent on your environment and the type of failure. You can check the time that the backup disconnected by searching for a message similar to the following in the cconsole.log file on the backup: MirrorClient: Primary AckDaemon failed to answer status request.
- The backup was considered active at the time that it detected failure. An active backup is one that is caught up and is receiving data synchronously from the primary. You can confirm this by searching for a message similar to the following in the cconsole.log file on the backup: Failed to contact agent on former primary, can't take over. Note: presence of a message like Non-active Backup is down in the cconsole.log file on the backup indicates that it was not active at the time of detecting failure. In this case, it can be assumed that the backup does not have all of the necessary data from the primary, and it is not safe to perform manual failover, as it will result in data inconsistency.

To continue with the manual failover, use the Force Become Primary function on the backup to become primary. This option is available from the ^MIRROR utility.

If you decide to proceed with the manual failover, and there is data inconsistency between the two failover members, the following consequences can be expected:

- Your application may be missing data that was committed on the former primary but never transmitted to the backup.
- The former primary will refuse to re-join the mirror as backup when it is restarted.
- You will need to restore a backup of your databases from the current (new) primary to the other system in order for it to re-join the mirror as a backup.