InterSystems®
Caché

Caché High Availability Solutions

Version 2018.1
2018-12-10

InterSystems Corporation   1 Memorial Drive   Cambridge MA 02142   www.intersystems.com

Caché High Availability Solutions
Caché Version 2018.1 2018-12-10

Copyright © 2018 InterSystems Corporation
All rights reserved.

InterSystems, InterSystems Caché, InterSystems Ensemble, InterSystems HealthShare, HealthShare, InterSystems TrakCare, TrakCare, InterSystems DeepSee, and DeepSee are registered trademarks of InterSystems Corporation.

InterSystems IRIS Data Platform, InterSystems IRIS, InterSystems iKnow, Zen, and Caché Server Pages are trademarks of InterSystems Corporation.

All other brand or product names used herein are trademarks or registered trademarks of their respective companies or organizations.

This document contains trade secret and confidential information which is the property of InterSystems Corporation, One Memorial Drive, Cambridge, MA 02142, or its affiliates, and is furnished for the sole purpose of the operation and maintenance of the products of InterSystems Corporation. No part of this publication is to be used for any other purpose, and this publication is not to be reproduced, copied, disclosed, transmitted, stored in a retrieval system or translated into any human or computer language, in any form, by any means, in whole or in part, without the express prior written consent of InterSystems Corporation.

The copying, use and disposition of this document and the software programs described herein is prohibited except to the limited extent set forth in the standard software license agreement(s) of InterSystems Corporation covering such programs and related documentation. InterSystems Corporation makes no representations and warranties concerning such software programs other than those set forth in such standard software license agreement(s). In addition, the liability of InterSystems Corporation for any losses or damages relating to or arising out of the use of such software programs is limited in the manner set forth in such standard software license agreement(s).

THE FOREGOING IS A GENERAL SUMMARY OF THE RESTRICTIONS AND LIMITATIONS IMPOSED BY INTERSYSTEMS CORPORATION ON THE USE OF, AND LIABILITY ARISING FROM, ITS COMPUTER SOFTWARE. FOR COMPLETE INFORMATION REFERENCE SHOULD BE MADE TO THE STANDARD SOFTWARE LICENSE AGREEMENT(S) OF INTERSYSTEMS CORPORATION, COPIES OF WHICH WILL BE MADE AVAILABLE UPON REQUEST.

InterSystems Corporation disclaims responsibility for errors which may appear in this document, and it reserves the right, in its sole discretion and without notice, to make substitutions and modifications in the products and practices described in this document.

For Support questions about any InterSystems products, contact:

InterSystems Worldwide Response Center (WRC)

Tel:	+1-617-621-0700
Tel:	+44 (0) 844 854 2917
Email:	support@InterSystems.com

Table of Contents

List of Figures

Caché High Availability Solutions

High availability (HA) refers to the goal of keeping a system or application operational and available to users a very high percentage of the time, minimizing both planned and unplanned down time. Caché provides its own HA solution, and easily integrates with common HA solutions supplied by operating system providers.

The primary mechanism for maintaining high system availability is called *failover*. Under this approach, a failed primary system is replaced by a backup system; that is, production *fails over* to the backup system. Many HA configurations also provide mechanisms for *disaster recovery* (DR), which is the resumption of system availability when HA mechanisms have been unable to keep the system available.

This article briefly discusses these general strategies for achieving HA with Caché-based applications:

- No HA Solution
- OS-Level Cluster HA
- Virtualization Platform HA
- Caché Mirroring

This article also discusses issues in Caché HA solutions, provides an HA solutions feature comparison, and discusses using Enterprise Cache Protocol (ECP) with a failover strategy.

1 Issues in Caché HA Solutions

Bear in mind the following two significant issues when evaluating potential HA solutions for your Caché systems:

- Shared storage

  An important principle of HA architecture is the avoidance of *single points of failure*. Most HA solutions rely on a shared storage component; which represents just such a risk; if the storage fails, it is impossible to keep the system available. Storage-level redundancy can mitigate this risk to an extent, but can also carry forward some types of data corruption.

  Caché mirroring, on the other hand, uses *logical data replication* between fully independent primary and backup storage, which eliminates the single point of failure problem entirely and avoids carrying forward most types of corruption.

  When using a solution other than mirroring, therefore, a single storage failure can be disastrous. For this reason, disk redundancy, database *journaling* as described in the “Journaling” chapter of the *Caché Data Integrity Guide*, and good backup procedures, as described in the “Backup and Restore” chapter of the *Caché Data Integrity Guide*, must always be part of your approach, as they are vital to mitigating the consequences of disk failure.

- Caché upgrades

  Many HA solutions allow for planned down time of a given component system without interrupting overall availability. Most, however, require significant down time to upgrade the production Caché instance.

  Caché mirroring, however, allows for minimum downtime Caché upgrade when application code, classes, and routines are kept in separate databases from application data; see Minimum Downtime Upgrade with Mirroring in the “Upgrading Caché” chapter of the *Caché Installation Guide* for more information.

HA solutions other than mirroring therefore require careful planning of down time windows for Caché upgrades, or any other maintenance requiring Caché shutdown.

2 No HA Solution

The structural and logical integrity of your Caché database is always protected from production system failure by the built-in features described in the "Introduction to Data Integrity" chapter of the *Caché Data Integrity Guide*: write image journaling, database journaling, and transaction processing. With no HA solution in place, however, a failure can result in significant down time, depending on the cause of the failure and your ability to isolate and resolve it. For many applications that are not business-critical, this risk may be acceptable.

Customers that adopt this approach share the following traits:

- Clear and detailed operational recovery procedures, including journaling and backup and restore.
- Disk redundancy (RAID and/or disk mirroring).
- Ability to replace hardware quickly.
- 24/7 maintenance contracts with all vendors.
- Management acceptance and application user tolerance of moderate downtime caused by failures.

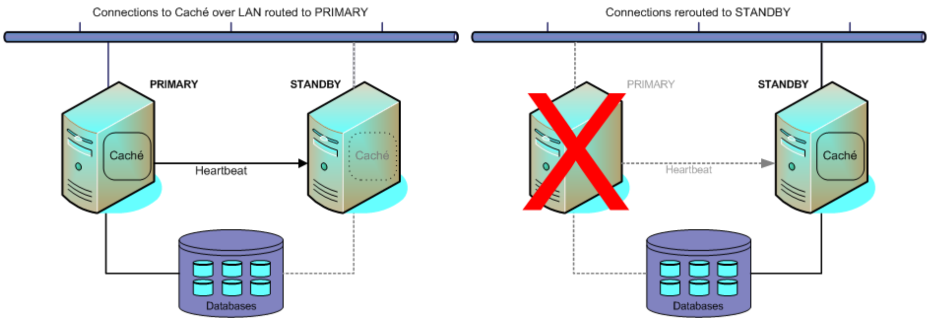
3 OS-Level Cluster HA

A common approach to achieving HA is the *failover cluster*, in which the primary production system is supplemented by a (typically identical) standby system, with shared storage and a cluster IP address that follows the active member. In the event of a production system failure, the standby assumes the production workload, taking over the programs and services formerly running on the failed primary, including Caché.

Caché is designed to integrate easily with failover solutions provided at the operating system level, such as Microsoft Windows Server Clusters, IBM PowerHA SystemMirror, Red Hat Enterprise Linux HA, and others. A single instance of Caché is installed on the shared storage device so that both cluster members recognize the instance, then added to the failover cluster configuration so it will be started automatically as part of failover. If the active node becomes unavailable for a specified period of time, the failover technology transfers control of the cluster IP address and shared storage to the standby and restarts Caché on the new primary. On restart, the system automatically performs the normal startup recovery, with WIJ, journaling, and transaction processing maintaining structural and data integrity exactly as if Caché had been restarted on the failed system.

The standby server must be capable of handling normal production workloads for as long as it may take to restore the failed primary. Optionally, the standby can become the primary, with the failed primary becoming the standby once it is restored.

Figure 1: Failover Cluster Configuration

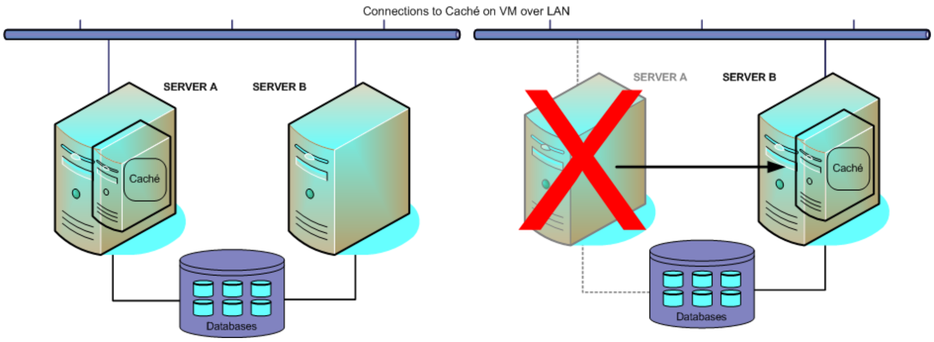


Under this approach, failure of the shared storage device is disastrous. For this reason, disk redundancy, journaling and good backup and restore procedures are critically important to providing adequjate recovery capability.

4 Virtualization Platform HA

Virtualization platforms generally provide HA capabilities, which typically monitor the status of both the guest operating system and the hardware it is running on. On the failure of either, the virtualization platform automatically restarts the failed virtual machine, on alternate hardware as needed. When the Caché instance restarts, it automatically performs the normal startup recovery, maintaining structural and logical integrity as if Caché had been restarted on a physical server.

Figure 2: Failover in a Virtual Environment



Virtualization HA has the advantage of being built into the virtualization platform infrastructure, and thus can require very little effort to configure, in some cases none at all. In addition, virtualization platforms allow the planned relocation of virtual machines to alternate hardware for maintenance purposes, enabling upgrade of physical servers, for example, without any down time.

5 Caché Mirroring

Caché mirroring with automatic failover takes a different approach to HA, relying on logical data replication between fully independent systems to avoid the single point of failure risk ofy shared storage and ensure that production can immediately fail over to an alternate Caché instance in almost all failure scenarios—system, storage, and network.

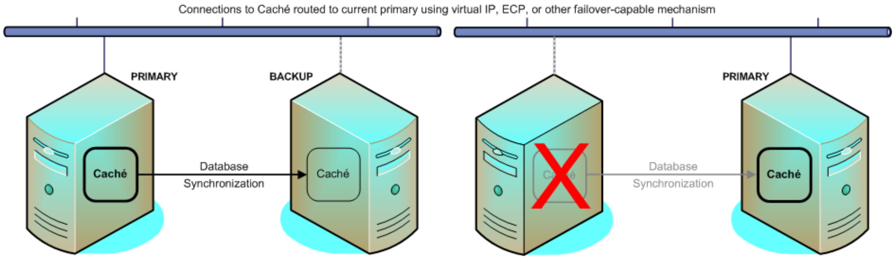
In a Caché mirror, one Caché instance, called the *primary failover member*, provides access to the production databases. Another instance on a separate host, called the *backup failover member*, communicates synchronously with the primary, retrieving its journal records, acknowledging their receipt, and applying them to its own copies of the same databases. In this way, both the primary and the backup always know whether the backup has the most recent journal files from the primary, and can therefore precisely synchronize its databases with those on the primary.

When this is the case, the mirror can quickly and automatically fail over to the backup in the event of a primary outage with no loss of data. A third system, the *arbiter*, helps the backup determine whether it should take over when the primary becomes unresponsive. A mechanism such as a virtual IP address shared by the failover members or Caché ECP redirects application connections to the new primary. The failover process takes just seconds; many users won't even notice it happening. And because the backup has its own copies of the databases, even a total failure of the primary and its storage does not render the databases unavailable. In fact, even when the backup is missing the most recent journal data, the backup's *mirror agent* can retrieve it from the primary's host, if it is still online.

Once you restore the former primary to operation following failover, it becomes the backup and its databases quickly catch up with those on the new primary, returning the mirror to full HA capability. You can then return the systems to their original roles ior maintain the new arrangement.

Mirrors can also include *disaster recovery (DR) async* members, which are asynchronously maintained copies of the primary; a DR async can be promoted to failover member, for example to become backup when a failed primary cannot be restored to operation quickly, or (if physically separate) for disaster recovery when an outage, such as a data center failure, brings down both failover members. Finally, mirrors can contain *reporting async* members, which maintain asynchronous copies of the production databases for business intelligence and data warehousing purposes.

Figure 3: Caché Mirror


Connections to Caché routed to current primary using virtual IP, ECP, or other failover-capable mechanism
PRIMARY
BACKUP
Caché
Database
Synchronization
Caché
Database
Synchronization
PRIMARY
Caché

Mirroring can also be used together with virtualization platform HA to create a hybrid HA approach, under which the virtualization platform responds to unplanned system or OS-level outages while mirroring handles all planned outages and unplanned database outages (including both Caché outages and storage failures) through automatic failover.

For complete information about Caché mirroring, including its DR capabilities, see the “Mirroring” chapter of the *Caché High Availability Guide*.

6 HA Solutions Feature Comparison

The following table provides a very general feature comparison mirroring, clustering, and virtualization as HA solutions.

	Caché Mirroring	OS-Level Clustering	Virtualization Platform HA
Failover after machine power loss or crash	Handles machine failure seamlessly in Caché version 2015.1 or later. Prior versions did not fail over automatically in this scenario; alternatives required careful planning.	Handles machine failure seamlessly.	Handles physical and virtual machine failures seamlessly.

	Caché Mirroring	OS-Level Clustering	Virtualization Platform HA
Protection from storage failure and data corruption	Built-in replication protects against storage failure; logical replication avoids carrying forward most types of corruption.	Relies on shared storage device, so failure is disastrous; storage-level redundancy optional, but can carry forward some types of corruption	Relies on shared storage device, so failure is disastrous; storage-level redundancy optional, but can carry forward some types of corruption.
Failover after Caché shutdown, hang, or crash	Rapid detection and failover is built in.	Can be configured to fail over after Caché outage.	Can be configured to fail over after Caché outage.
Caché upgrades	Allows for minimum- downtime Caché upgrades.*	Caché upgrades require downtime.	Caché upgrades require downtime.
Application mean time to recovery	Failover time is typically seconds.	Failover time can be minutes.	Failover time can be minutes.
External file synchronization	Only databases are replicated; external files need external solution.	All files are available to both nodes.	All files available after failover.

* Requires a configuration in which application code, classes, and routines are kept in databases separate from those that contain application data

7 Using ECP with a Failover Strategy

Whatever approach you take to HA, Enterprise Cache Protocol (ECP) can be used to provide a layer of insulation between the users and the database server. Users remain connected to ECP application servers when the database server fails; user sessions and automatic transactions that are actively accessing the database during the outage will pause until the database server becomes available again through completion of failover or restart of the failed system.

Bear in mind, however, that adding ECP to your HA strategy can increase complexity and introduce additional points of failure.

For information about ECP features and implementation, see the “ ECP Failover” chapter of the *Caché High Availability Guide* and the *Caché Distributed Data Management Guide*.