Caché Upgrade Checklists

Version 2009.1
06 August 2009

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

Caché Upgrade Checklists
Caché Version 2009.1 06 August 2009

This book was assembled and formatted in Adobe Page Description Format (PDF) using tools and information from the following sources: Sun Microsystems, RenderX, Inc., Adobe Systems, and the World Wide Web Consortium at www.w3c.org. The primary document development tools were special-purpose XML-processing applications built by InterSystems using Caché and Java.

For Support questions about any InterSystems products, contact:

InterSystems Worldwide Customer Support

Tel:	+1 617 621-0700
Fax:	+1 617 374-9391
Email:	support@InterSystems.com

Table of Contents

List of Tables

About This Book

This book provides information on changes in Caché that can affect the upgrade of existing systems and applications to newer versions. It is intended for system administrators, application developers, and operators.

READ THIS INFORMATION BEFORE UPGRADING.

This book contains the following sections:

- General Upgrade Information provides information pertinent to all upgrades, and the specific release checklists
- Checklist for 2009.1
- Checklist for 2008.2
- Checklist for 2008.1
- Checklist for 2007.1
- Checklist for 5.2
- Checklist for 5.1

Each of the release-specific checklist provides information on moving to that release from the previous release and a more detailed Table Of Contents.

The following books provide related information:

- Caché Release Notes lists new features in recent Caché releases.
- Supported Platforms lists the platforms supported by this release of Caché.
- Caché Installation Guide describes the process of installing Caché on your system.
- Introduction to Caché provides an overview of the features and major components of Caché.

1

General Upgrade Information

Background

Caché version 5.1 was a significant improvement in functionality and security over its predecessors. In making this advance, InterSystems goal was to provide a compatible evolutionary path forward whenever possible. However, many of the features, such as the System Management Portal, replaced functions in previous releases with new mechanisms. Furthermore, the addition of the new security features required a partial redesign and reorganization of the underlying system. These introduced incompatibilities with previous versions of Caché.

Documenting these changes in a single place proved useful and has been continued for subsequent releases of Caché.

Organization

The major divisions of this document represent the significant changes from the prior release, starting with the current version. Within each release the material is organized by whether it primarily affects administrators, operators, or application designers/developers of existing systems. The divisions are not meant to be exclusive; many of the items of interest to administrators also affect application developers. Readers are encouraged to review all pertinent material in other sections.

1.1 Upgrading Caché

1.1.1 Upgrading from Field Test Versions

CAUTION: InterSystems does not support an upgrade from any Field Test version to another Field Test version, nor from a Field Test version to an officially released version. The term "Field Test version" includes any version of Caché labelled as such, or distributed to customers at DevCon or other events.

1.1.2 Upgrading From Prior Released Versions

Customers running on any prior released version of Caché may upgrade to this version of Caché during installation. When upgrading across multiple versions, intermediate upgrade steps may be necessary depending on the inter-release compatibility requirements. The release notes for the intervening releases will contain that information.

After each upgrade step, the following conditions apply:

- Routines

  Compiled routines are upward-compatible from version to version and do not need to be recompiled unless specified otherwise in the upgrade notes for a specific release.

  Because most versions contain improvements in the routine compiler, in the generated code, and/or efficiencies in the runtime support, customers may decide to recompile their routines to take advantage of new features and meta-routine information. Those customers whose actions implicitly access this underlying information must recompile their routines. An example of such access is source-level debugging on version X of code compiled on version earlier than X.

  If you recompile routines, they should be recompiled after classes are recompiled.

- Classes, SQL, Projections, Proxy Classes

  InterSystems requires that all classes be recompiled after an upgrade. The following command will upgrade and compile the class dictionaries in every namespace:

  ```
  Do $SYSTEM.OBJ.CompileAllNamespaces("u")
  ```

  You must regenerate any proxy classes used in the upgraded instance by following the instructions in the appropriate guide(s) in the *Caché Language Bindings* set. You must also purge any cached queries in any upgraded namespace by issuing the command

  ```
  Do $SYSTEM.SQL.Purge()
  ```

  in that namespace.

- Exported XML

  Unless specified in the upgrade notes for a specific release, exported XML files can be imported into later releases.

  The reverse is NOT true. Later versions could be using new features not present in earlier versions, and could have a different class dictionary version number which alters how these classes are stored internally that can not be undone when importing to a previous version.

- Debugging

  InterSystems also recommends recompiling routines and classes for applications under development. This synchronizes the debugger with the expected format of the compiled code.

1.1.3 Upgrading ECP Configurations

The following guidelines apply to the process of upgrading ECP configurations:

- Unless explicitly noted for a particular version, the protocol used to communicate among ECP systems is fully compatible across versions.
- For ECP configurations consisting of stateless, independent database and application servers, an ECP configuration may be updated incrementally, system by system.

  The application servers should be upgraded to the new version first. Once all application servers have been upgraded, the database servers can be serially upgraded.

If your database or application servers hold local transaction information until the transactions are completed, or if you have questions or concerns about how to upgrade your ECP configuration, please contact InterSystems Worldwide Customer Support.

2

Caché 2009.1 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2009.1 that, because of their difference in this version, affect the administration, operation, or development activities of earlier systems.

Those customers upgrading their applications from earlier releases are strongly urged to read the upgrade checklist for the intervening versions as well. This document addresses only the differences between 2008.2 and 2009.1.

The upgrade instructions listed at the beginning of this document apply to this version.

2.1 Administrators

This section contains information of interest to those who are familiar with administering prior versions of Caché and wish to learn what is new or different in this area for version 2009.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

2.1.1 CPF File Changes

Significant changes have been made to the format and content of the configuration parameter file (CPF). In addition, more stringent validation of the file is done during system startup. Valid files from earlier releases will be automatically converted to the new format upon first upgrade to 2009.1.

WARNING! Invalid CPF files will prevent Caché from installing or starting up. The currently active CPF file must be validated, and errors fixed, BEFORE the system is upgraded to this version. Validation and correction of the CPF file should occur a few days before the system is upgraded, and rechecked just before upgrade to make sure any issues are addressed in advance of the upgrade.

There are two ways to validate the CPF file before upgrading a particular system:

1. For those customers who subscribe to the Technical Assistance program, InterSystems provides a standalone program to check the validity of CPF files before installing 2009.1 to valid customers. The program is available from the WRC Direct page. Once you login, choose **CPF Validation** from the available options.

   You will be prompted to enter the directory and filename of the CPF file you want to validate. The file will be uploaded to the WRC, validated, and a list of any errors will be displayed. You can then correct any errors, and revalidate the file until all errors are corrected.

   If you do not have a userid and password, contact InterSystems Worldwide Customer Support.

2. From a system where 2009.1 is already installed, start a terminal session. From the terminal session, in the %SYS namespace, you can run the validation routine as follows:

   ```
   Set Status = ##Class(Config.CPF).Validate(<CPFFILE>)
   ```

   where <CPFFILE> is the directory and filename of the currently active CPF file. A list of errors will be displayed on the terminal screen. You can then edit the CPF file, correct the errors, and rerun the validation routine.

In both cases, after all corrections have been made, and the CPF file has been completely validated, you can replace the CPF file and restart your system to have the new settings take effect.

If you use any additional CPF files other than the currently active one, you must also run the validation routine and correct any errors against those CPF files before they can be used.

2.1.2 Changes In Package, Routine, And Global Mappings

In version 2009.1, two major changes have been made to the way globals, routines, and global subscripts are mapped:

1. The rules for these items have been unified.
2. Erroneous mapping definitions now result in configuration file validation errors that prevent Caché from starting up.

In prior versions, global subscript mappings could not have overlapping ranges such as

```
A("A"):A("D")->DB1
A("B"):A("E")->DB2
```

but this was not true of global and routine mappings. In these instances, the mappings were applied in the order found in the configuration file and errors were silently reconciled. For example, a mapping like

```
A:D ->DB1
B:E ->DB2
```

would have been silently interpreted as the mapping

```
A:D -> DB1
D:E -> DB2
```

when the system was started.

Note: The range "A:D" is treated as the mathematically half-open interval [A-D), that is, beginning with "A" up to, but not including, "D" .

In prior versions, the order of the mappings was important. In the conversion of the mappings for use in 2009.1, any overlaps are converted to an equivalent set of non-overlapped mappings. The following table gives some examples:

Prior Version Mapping	Converted Mapping For 2009.1
B* –> SAMPLES A:G –> USER	A:B –> USER B:C –>SAMPLES C:G –> USER
B:C –> SAMPLES A:G –> USER	A:B –> USER B:C –>SAMPLES C:G –> USER
A –> SAMPLES A:G –>USER	A –>SAMPLES B:G –> USER
A* –> SAMPLES A:G –>USER	A:B –>SAMPLES B:G –> USER
A:G –> USER B:H –> SAMPLES	A:G –> USER G:H –> SAMPLES

Because the conversion process produces non-overlapped mappings, the order of the mapping is no longer important.

In 2009.1, overlapped mappings can still be present in the CPF file or entered into the Management Portal as long as they form proper subsets. This means that mappings such as those in the first column of the last row of the table will be converted as shown when upgrading to version 2009.1. However, this same set will result in errors if found once the system has been upgraded.

Important: Before upgrading a system from 2008.2 to 2009.1, sites are strongly urged run the CPF validator to determine if the configuration file it intends to use is correct. The ways to do this were described in a preceding section.

Sites that have overlapped mappings are strongly encouraged to test the converted mappings to make sure that all previous data remains accessible and that new data is stored where it was intended.

It is good practice to validate all the CPF files a site intends to use before they are first employed in production. If a site switches to an alternate CPF file without converting it, the CPF file will be converted to the new format when it is used for the first time.

If you need assistance with the conversion process, please contact InterSystems Worldwide Customer Support.

Management Portal Changes

Because of the new rules prohibiting overlaps, the order of the mappings in a configuration file no longer matters. For this reason,

- Mappings will be displayed in the Management Portal in the order of the collation for the current locale.
- The **move down** and **move up** buttons have been removed because they no longer apply.

2.1.3 Version Interoperability

A table showing the interoperability of recent releases is now part of the Supported Platforms document.

2.1.4 Management Portal Changes

Numerous changes have been made in the Management Portal for this release both to accommodate new features and to reorganize the existing material to make it easier to use. Among the more prominent changes are:

- The underlying algorithms used to map globals, global subscripts, routines and packages have changed, and the pages that deal with these have changed accordingly.
- The decision to deprecate DCP and DDP in a future release caused the pages that refer to these to be removed, and the corresponding sections to be removed from the cpf file. Administrators must now configure these networking facilities via the ^LEGACYNETWORK utility.

2.1.5 Operational Changes

2.1.5.1 Journal Wait Time Extended For System Shutdown

During shutdown, the system tries to wait for the end of the journal file to appear in the Write-Image Journal (WIJ) recovery information. When this happens, the system can remember its shutdown state as "no journal recovery required at startup". This change extends the time allowed for that to happen from one minute to 30 minutes, if the write daemon is busy writing blocks to disk. This doesn't change the behavior of shutdown, only the amount of time taken. On clustered systems, when the write daemon exits and no recovery is required, the daemon removes the journaling information from the WIJ as a signal that no recovery is required.

2.1.5.2 Mounting Databases At Caché Startup

In version 2008.2, Caché would attempt to mount all databases. Those having the "mount required at startup" set to YES would cause startup to fail if Caché could not mount them.

In 2009.1, the behavior for databases required at startup is the same. However, other databases will not be mounted until they are first accessed. This is most visible in the Management Portal right after Caché begins execution.

Note: Caché distinguishes between "dismounted" databases which have been made inaccessible to instances, and "unmounted" databases which are available and will be mounted the first time they are accessed.

2.1.5.3 Audit Database Changes

Index Record Changes

An index on timestamps has been added to improve query performance in the audit database. Previously, the audit data was not indexed by datetime. When Caché is restarted following an upgrade to 2009.1, the existing audit database will be converted to the new format by a background job which is launched during startup.

As a result of this change, the format of the audit record global has been changed. Previously, it was:

```
^CacheAuditD(<System>, <ID>) = $lb(<AuditDdata>)
```

The new format is:

```
^CacheAuditD(<UTCTimeStamp>, <System>, <ID>) = $lb(<AuditDdata>)
```

If the site has copied audit data to a different namespace for archival and reporting, that audit data will be converted the first time it is accessed by the **^%AUDIT** routine for reporting.

If an application only uses SQL to report on an archived audit database, then it must be changed to do one of two things:

1. Run **^%AUDIT** in the namespace where the data is archived and run a list report on the data. This will cause the conversion of the data.
2. Run the **##Class(%SYS.Audit).Convert()** method in the namespace where the data is archived.

See the %SYS.Audit class for more details on the audit record format.

Audit Entries No Longer Mandatory

In prior versions, some system level audit events used to be marked as *mandatory* audit events. This meant that an administrator could not turn off auditing for these events. In 2009.1., Caché now allows any auditing event to be disabled. By default, audit entries that were marked as mandatory in previous releases remain on and must be explicitly turned off.

Note: Queries in the Security.Events class have been updated to remove the "Mandatory" column from the query. Customers using these queries may need to update their application to remove these columns.

2.1.5.4 NetWide Domain Namespaces (NWDS) Removed

Beginning with 2009.1, the Netwide Domain Spacespaces (NWDS) feature is no longer supported. Customers who have relied on this to propagate namespace changes across system boundaries should contact InterSystems Worldwide Customer Support for guidance.

2.1.6 Platform-specific Items

This section holds items of interest to users of specific platforms.

2.1.6.1 No Support For CPUs Before Pentium P4

Beginning with this version, Caché will only install and run on Intel-based platforms using the Pentium P4 or later chipset, that is, those that support the SSE2 cpu extensions. This also applies to other manufacturers equivalent chipsets, for example, those from Advanced Micro Devices (AMD) that are functionally equivalent to the Intel Pentium P4.

Note: In version 2008.1, this restriction only applied to server systems. Now it is applicable to BOTH client and server installations.

2.1.6.2 Tru64 UNIX Version And Configuration Specifics

Patch Level

InterSystems recommends that Tru64 UNIX systems be upgraded to a minimum level of 5.1B-4. This contains a correction that corrects "... a problem with pagetable page allocations that could leave a thread waiting indefinitely during a fork operation." according to the vendor.

Stack Size Limit

Some systems running Tru64 UNIX may display messages from the operating system that indicate the stack cannot be increased in size. The appearance of this message depends on the configuration parameters and application load. More space can be allocated to the stack through the **ulimit** command.

MultiValue Queries

MultiValue does not support queries on Tru64 UNIX under this version of Caché.

2.1.6.3 Microsoft Windows

Default Windows DSN Now ODBC 3.5 Compatible

The default SAMPLES and USER DSN created during a Windows installation will use the InterSystems ODBC35 driver. This is true for new installs and upgrades from any released version. If a customer application uses default DSN created by a Caché install, it now must ODBC 3.5 compatible, or the application must be changed to reference a specific, compatible DSN.

ODBC Driver Version Management Changed With New Installer

With the new Windows installer, ODBC driver kits overwrite files in the common directory which have the same version as those in the kit. Prior to this version, kits were only overwriting files which had a lower version number. To enable old behavior, the following command line option can be used:

```
ODBCDriver_x86.exe /V"REINSTALLMODE=omus"
```

To make ODBC kit overwrite any files regardless of version, the following command line option is used:

```
ODBCDriver_x86.exe /V"REINSTALLMODE=amus"
```

Details on installing Caché via a command line are given in the technical article on Windows Silent Installation.

Windows Errors In New Installer Upgrading Older Installations

When the new MSI-based install upgrades a non-MSI instance, rollback will be disabled. If the installation process encounters a problem, no files will be deleted. This behavior is similar to upgrades in the old-style installations. Normally repair should fix the instance after error is corrected.

Rollback is still enabled for new installs and upgrades over MSI instances. In this case, the system will be restored to the original state if an error is encountered during install or upgrade.

2.1.6.4 Weblink No Longer In Distribution Kit

Starting in 2009.1, Weblink is no longer packaged in the Caché installation kits. The latest Weblink packages for each platform are available free of charge to supported customers. They can be downloaded from the Worldwide Response Center distribution page.

These kits contain all the software components and documentation needed to install and use WebLink. Instructions for installing Weblink can be found in the documentation subdirectory of the Weblink kits. If you have any questions, please contact the Worldwide Response Center.

2.2 Developers

This section contains information of interest to those who have designed, developed and maintained applications running on prior versions of Caché.

The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

2.2.1 General Operational Changes

2.2.1.1 Change Timestamp Update For Routine Import

When importing a routine that does not have a timestamp either in XML format, or in %RO format with %apiRTN, if this routine is determined to be identical to the one already in the system, Caché will not change the last-modified time of the routine.

2.2.1.2 Global Name Truncation Changed

When a global name was truncated to the maximum length of 31 characters, an invalid name would result if the 31st character is a period. Trailing periods will now be removed when the name is truncated.

Prior to this change, application with names of this form would run without error, even though an attempt to access the global by ZWRITE, ^%GD, ^INTEGRITY, and so on would fail. With this version, the application will continue to run but will be using a different-truncated global name. In the case of an Caché upgrade, any existing data will be invisible to it.

2.2.1.3 %BuildIndices Now Includes Bitmap Extent By Default

The default list of indices built by **%BuildIndices** now includes the bitmap extent index, if it exists and has not been previously built. In previous versions, the bitmap extent index would only be automatically included if other bitmap indices were also built and the bitmap extent index had not been previously built.

2.2.1.4 TROLLBACK Failure Changes

This version of Caché changes what happens when a transaction fails to roll back receiving a <ROLLFAIL> error. The new behavior is based on the setting of the system flag "Freeze on Journal Error" and proceeds as follows for local (that is, non-ECP) transactions:

1. If the system flag is NOT set, the process initiating the transaction and TROLLBACK gets a <ROLLFAIL> error. The transaction is closed. Locks retained for the transaction are released. This option trades data integrity for system availability.
2. If the system flag is set, the initiating process halts and clean job daemon (CLNDMN) retries rolling back the open transaction. During the CLNDMN retry period, locks retained for the transaction are intact and as a result the system might hang temporarily. This option trades system availability for data integrity.

Important: For transactions on ECP configurations, it is the system flag on the ECP server that governs the behavior of the TROLLBACK failure. When the system flag on the ECP server is set, the ECP client process initiating TROLLBACK hangs (unlike the local case), while the ECP worker daemon on the server retries rolling back the open transaction. The remainder of the operation is similar to the local case.

Note: If journaling is disabled within the process (for example, by invoking DISABLE^%SYS.NOJRN), at the time of issuing TROLLBACK, journaling will be enabled for the duration of TROLLBACK and then disabled upon exiting TROLLBACK. Previously, one would get a <ROLLFAIL> error in this case.

2.2.1.5 Streams Default Directory Is Now Per-Namespace

In version 2009.1, streams directory no longer defaults to the Caché temp directory. Instead, the default directory is a per-namespace value with a different location possible for each namespace. The default location is now defined as:

- For ECP databases, the default will still be the directory holding CACHETEMP.
- Otherwise, streams will be stored in a subdirectory named, stream, under the default directory for the globals of that namespace. For example, the default directory for streams in the SAMPLES namespace is <install-dir>/mgr/samples/stream.

The default location can be changed programmatically by setting the value of *StreamLocation* via the Config.Databases class for a particular namespace.

2.2.2 IPv6 Changes

The ability of Caché to support IPv6 addressing and networks is an important addition to the capabilities of this version. However, IPv6 represents network addresses in a new and expanded format. Applications that expect to receive and parse the format of IP addresses will likely have to be rewritten to handle the both IPv4 and IPv6 addresses.

2.2.2.1 TCP Device Representation Changes

When IPv6 is enabled, the network address returned by Caché has the port number separated from the network address by a comma (,). This is because IPv6 addresses use the IPv4 separator, a colon (:)< as a part of the network address itself.

2.2.2.2 Wildcards Not Allowed In Selections

Caché does not support the use of a wildcard character such as "*" in an IPv6 address. In situations where one is chosen as a filter, for example, to accept or reject a connection, you must specify a host-name or specific address.

2.2.2.3 Caché Direct Connection String Issues

IPv6 address formats are fully supported in Caché Direct. In particular, the Server/MServer property of the VisM.ocx allows IPv6 addresses within the same general connection string format as previously. However, depending on how they are used by an application, there may be some issues that need to be confronted. Specifically, the format of the connection string in Caché Direct is a colon-delimited expression, of the general form

```
CN_IPTCP:server_address[port]
```

where server_address is the master server and can be an IP address, or a server DNS name, or the special name "localhost".

Potential confusion arises with IPv6 addresses which contain colons in the server_address part. (IPv6 also supports more than one form of loopback address.) Caché Direct supports all of these format variations. But, depending on the assumptions made by application code, the new addresses could cause incorrect behavior if not parsed properly.

2.2.3 Objectscript Changes

2.2.3.1 Floating-point Number Rounding Improved

This version of Caché fixes a deficiency in the algorithm used to round **$DOUBLE** values to a particular number of decimal digits. The new algorithm increases the accuracy of this rounding and eliminates overflows that occurred in some intermediate steps.

For example, consider the expression

```
$NUMBER($DOUBLE(123456789.0123456) ,3)
```

that requests the value be rounded to 3 decimal places. In previous versions, the result would be 123456789.01200000941. In this version, the value returned is 123456789.01199999451. Neither of these are the expected, 123456789.012.

The reason for this is that the expected answer has no exact representation in binary floating-point. In 2009.1, the rounding algorithm chooses the an exact floating-point value which is closest to the expect value.

CAUTION: This change means that some values may differ from previous versions when rounded to a specific number of places such as xDBC conversions to numeric values with a fixed scale.

Caché has also improved the rounding behavior in the kernel with the following effects:

- If the conversion from a **$DOUBLE** value to decimal digits can have more than 38 significant digits, the result is rounded at the 38th digit and all following digits will be zeroes.
- Under some circumstances in previous versions, the converted value did not have the correct number of trailing zeroes; this is now corrected.
- In earlier versions, the low order digit of the result might be rounded differently. In this version, it is rounded to the closest representable value (that is, more accurately).
- If all the significant digits of a **$DOUBLE()** value are beyond the scale position, the resulting digits will all be 0 (a minus sign will be included if the number were negative before scaling). However, if all the significant digits of a **$DECIMAL()** value are beyond the scale position, the resulting digits will all be 0 and there will be no indication of a negative result.

  This behavior has not changed from earlier releases. What is different is that **$JUSTIFY** will always format a **$DOUBLE(-0)** with a minus sign and that **$FNUMBER** will format **$DOUBLE(-0)** with a minus indication when the format string includes a "D" flag.

2.2.3.2 Remove $ZUTIL(69, 4, ...)

This $ZUTIL function originally made the principal device of the child job the same that is used by the parent. It has been non-functional since version 5.1. It has been removed from this and future versions of Caché; the documentation has also been removed. Attempts to use this entrypoint will result in a <FUNCTION> error.

2.2.3.3 ZINSERT Or ZREMOVE On Current Routine No Longer Allowed

ZINSERT and ZREMOVE can not be used to modify the running routine. If an existing application attempts this, an <EDITED> error will be thrown when control returns to the modified routine. In prior

versions, the modification was not properly detected resulting in a process access violation. The proper usage for these commands from a routine is within an XECUTE and preceded by a ZREMOVE or ZLOAD command.

2.2.4 Routine Compiler Changes

2.2.4.1 Macro Preprocessor Corrections

In previous versions, the compilation of the following statement:

```
#dim X as %String // Length = 10
```

would incorrectly result in a line

```
Set X=10
```

Now the comment symbol, "//" is handled correctly and no code is generated. Also, the macro pre-processor now correctly recognizes when a statement that looks like a pre-processor directive is inside a comment. So the following will no longer compile:

```
#include %occInclude
      /*
        A comment
#IF 0
        Old comment
      */
#ELSE
        New comment
      */
#ENDIF
```

2.2.5 Class Changes

2.2.5.1 Improved Signature Checking

In version 2008.2, signature checking in subclasses applied to all signature errors in a class. However, in some cases not all errors were reported. In version 2009.1, the compiler will now report all signature errors so they can all be fixed at once.

2.2.5.2 Size Field In FileSet Query

The *Size* property returned by the %File:FileSet query is changed from %Integer to %BigInt, It is possible for a file to be larger than 4GB in size.

2.2.5.3 Attributes In %Dictionary.ClassDefinition Can Now Be Undefined

Caché 2009.1 now allows attributes used in programmatically defining classes to be in an undefined state. Prior to this, unreferenced attributes were assumed to have their default value. Now, for each

attribute in %Dictionary.ClassDefinition, there are new methods, **<AttributeName>Reset()** and **<AttributeName>IsDefined()**.

- **<AttributeName>Reset()** will cause the attribute to be undefined. An undefined attribute will return the default value on a call to **Get**.
- Setting an attribute to any value will cause the attribute to be defined.
- **<AttributeName>IsDefined()** returns the defined state of its attribute.
- An attribute will be used to define the resulting class only if the attribute is defined.

Note: This applies only to the class, %Dictionary.ClassDefinition itself, and not for the member definition classes.

2.2.5.4 XML DOM Internal Representation Changed

The Caché data structure used to represent an XML DOM has changed. The details are defined as being internal and subject to change without notice. However, if an application is accessing this data directly, it must be changed and recompiled. The recommended way to access this data is via the %XML.Node class.

2.2.5.5 Prevent IDKEY Value Change On UPDATE

The value of an ID cannot be altered once assigned. Documentation does specify that ID values cannot be changed, but the software did not properly enforce the rule. This deficiency has been fixed and the IDKEY value cannot be updated.

2.2.5.6 SYS.DataBase.Freespace Query Changed

Applications that use the SYS.Database.Freespace query directly, and reference the "% Free" data column by name in the result set, will need be changed to reference the column by number, or by using the new label “Free” . The previous label interfered with proper XML generation.

2.2.5.7 Generate Cursor Name From Incremented Counter, Not Class Query Name

The cursor name generated to implement a class query of type %Library.SQLQuery now is based on an incremented counter instead of the query name. This resolves a <LABELREDEF> error when the query name is not unique in the first 29 characters.

2.2.5.8 GUID Management Changes

If a persistent class is marked as GUIDENABLED, Caché will assign Globally Unique IDentifiers (GUIDs) to each object when it is created. A later call to delete the object via **%Delete** on an object

will no longer delete the GUID for that object. There is history in the GUID global in each namespace where a GUIDENABLED object has been created. Users are responsible for removing entries from ^OBJ.GUID that are no longer needed.

Also, a new argument is now defined for the **%LIbrary.Persistent.GUID()** method that returns the GUID value for an OID. The argument, *pDeepSearch*, if true, will trigger a search with the extent and any registered subextents for an object that no longer exists in the database. Previously, %OnDetermineClass would fail for a deleted object and the GUID would not be found. Now, if *pDeepSearch* is true, the extent of the current class and all registered subextents are searched for the ID.

2.2.5.9 %Net.MailMessage And %Net.MailMessagePart Now Subclass %SerialObject

Before Caché 5.1 and Ensemble 4.0, the classes %Net.MailMessage and %Net.MailMessagePart were subclasses of %Library.SerialObject. Beginning with Caché 5.1 and Ensemble 4.0, they were changed to be subclasses of %Library.RegisteredObject. In this version of Cache and Ensemble, %Net.MailMessage and %Net.MailMessagePart are again subclasses of %Library.SerialObject.

This change allows archived data from releases earlier than Caché 5.1 and Ensemble 4.0 to be accessed by current applications.

2.2.5.10 Deprecated Classes

%Library.Float

This class is included only for purposes of backward compatibility. In the future, applications should use either

- the %Library.Double datatype class for cases where values are in the form of IEEE float (that is $DOUBLE) format, or
- the %Library.Decimal class for cases where values are in the form described by Caché $DECIMAL.

Config.Configuration

The Config.Configuration class is deprecated beginning with this version of Caché. Applications using it should be rewritten to use other, more specific classes in this package, for example, Config.Databases, Config.Devices, or Config.Telnet. Config.Configuration will be removed in a future release.

2.2.5.11 %Net.LDAP

The %NET.LDAP class is deprecated beginning with this version of Caché. Applications using it should be rewritten to use the equivalent classes in the %SYS.LDAP package.

2.2.5.12 Class Deletions

The classes listed within the following packages were present in 2008.2 and have been removed in version 2009.1:

- Package %Monitor.System – Database, Sample.Database
- Package %Net.SSH – , Session
- Package %SYS.Task – FreeSpace
- Package %Studio.Fireball – Error, Login, Namespaces, Probe
- Package %TSQL.Compiler – COS, Dynamic, Flo, ParseTree, Reflector, Visitor, sysSymbol

2.2.5.13 Class Component Deletions

The following class components, present in version 2008.2, have been removed from use in this version:

- %Activate.UI.Wizard

  Method: %OnSubmit, DrawTitle, Process, cbSelect, finish

  Parameter: APPLICATION
- %CPT.CalloutShell

  Method: cmdH
- %CSP.UI.Portal.NLS

  Method: DrawLocale
- %CSP.UI.System.MappingsAPI

  Method: CopyNamespaceMappings, GetTotalNumber, GlobalMappings, PackageMappings, RoutineMappings, UpdateDelete, UpdateMove
- %CSP.UI.System.MappingsTablePane

  Method: IsMappingModified, MappingInfoClose, MappingInfoExecute, MappingInfoFetch

  Query: MappingInfo
- %Debugger.System

  Method: Zbreak
- %Library.EnsembleMgr

  Method: activateConfiguration, openConfiguration
- %MV.StudioRoutines

  Method: Search, checkMatch

- %Monitor.System.Dashboard

  Method: GetAppErrs, GetECP, GetSMH, GetShadows, GloStats

- %Net.abstractMQ

  Property: Password, Username

- %SOAP.Security.BinarySecurityToken

  Property: Id

- %SOAP.Security.UsernameToken

  Method: Id

- %SYS.LDAP

  Method: Test

- %Studio.ClassMgr

  Method: ClassListClose, ClassListExecute, ClassListFetch, GetDefintion

  Query: ClassList

- %Studio.Debugger

  Method: Zbreak

  Property: PID

- %Studio.Project

  Method: ExecuteCommand, checkMatch

- %UnitTest.Manager

  Property: TheLog

- %UnitTest.SQLRegression

  Method: OnAfterAllTests, OnBeforeAllTests

- %XML.Security.KeyInfo

  Property: Key

- %XML.Security.Transform

  Property: content

- %ZEN.Report.aggregate

  Method: method

- Config.Configuration

Index: NameIndex

Method: %OnAfterSave, %OnBeforeSave, %OnDelete, %OnNew, %OnOpen, CPFExport, CPFImport

Query: List

- SYS.Cluster

  Method: IsMember

2.2.5.14 Class Method Signature Changes

The following methods have been incompatibly changed in either the parameter list needed to invoke them, or in the type of the value they return:

- %Activate.TLEnumerator

  Argument list: LoadTypeLibrary

- %Activate.UI.Wizard

  Argument list: SetSelected

  Return value: SetSelected

- %CPT.CalloutDump

  Argument list: DumpParseTree, DumpParseTreeNode, DumpParseTreeNodeAnn, DumpParse-TreeNodeChild, DumpParseTreeNodeMap, DumpParseTreeTop, ListParseTree

  Return value: DumpParseTreeNode, DumpParseTreeNodeAnn, DumpParseTreeNodeMap, DumpParseTreeTop, ListParseTree, ListParseTreeNode

- %CPT.CalloutIndex

  Argument list: BuildIndicesIfNeeded

- %CPT.CalloutShell

  Argument list: ParseLine, Run, cmdC, cmdR

- %CPT.CalloutTesting

  Argument list: Test, TestRoutine, TestStream

  Return value: Test, TestRoutine, TestStream

- %CPT.COSCallout

  Argument list: Compile

- %CSP.StreamServer

  Argument list: FileClassify

- %CSP.UI.Portal.NLSEdit

  Argument list: EditIOTable

  Return value: EditIOTable
- %CSP.UI.SQL.UserPrivPane

  Argument list: RevokeRow
- %CSP.UI.System.LicensePane

  Argument list: Save
- %CSP.Util.AutoFormDynamic

  Argument list: ProcessSubmit
- %Compiler.Util.Flo

  Argument list: addNode, pushNode
- %Dictionary.ClassDefinition

  Argument list: NameSet
- %Exception.AbstractException

  Argument list: OutputToStream
- %IO.I.Stream

  Argument list: ReadUntil, ReadUntilArray
- %Installer.Installer

  Argument list: ActivateConfiguration
- %Library.EnsembleMgr

  Argument list: createMappings, deleteMappings
- %Library.File

  Argument list: SubDirectoryName
- %Library.FileStreamAdaptor

  Argument list: GetStreamIdForFile
- %Library.JGWResultSet

  Argument list: CreateStaticRS
- %Library.Persistent

  Argument list: %GUID

- %Library.ProcedureContext

  Argument list: NewResultSet

- %Library.RoutineMgr

  Argument list: CompileClass

- %Library.SyntaxColor

  Argument list: Color, GetCSS

- %Monitor.Alert

  Argument list: GetId

  Return value: Create, Delete

- %Monitor.Manager

  Argument list: StopSystemCounters

  Return value: Activate, Alert, AppNotifyMethod, AppSmtpPassword, AppSmtpServer, AppSmtpUserName, ClearSystemCounters, Deactivate, Halt, HaltApp, Purge, Refresh, RefreshApp, SignalApp, SmtpPassword, SmtpServer, SmtpUserName, Start, StartApp, StartSystemCounters, StopSystemCounters

- %Projection.AbstractProjection

  Argument list: CreateProjection, RemoveProjection

- %SOAP.WSDL.Reader

  Argument list: Process

- %SOAP.Security.SecurityTokenReference

  Argument list: GetX509DirectReference

- %SYS.Audit

  Argument list: Exists, Get, Modify, OpenAuditItem, SearchExts

- %SYSTEM.Error

  Argument list: %OnNew

- %SYSTEM.License

  Argument list: DecodeAuth, SetUserLimit, Upgrade

  Return value: Upgrade

- %SYSTEM.OBJ

  Argument list: Upgrade, UpgradeAll

- %SYSTEM.SQL

  Argument list: SetDefaultSchema
- %Studio.AbstractDocument

  Return value: CompileDocument
- %Studio.ClassMgr

  Argument list: SaveDefinition
- %Studio.Debugger

  Return value: IsStopped
- %Studio.Project

  Argument list: searchClassNode, searchItem
- %TSQL.Transformer

  Return value: Evaluate
- %XML.Node

  Argument list: AppendChild
- %XML.Reader

  Argument list: Correlate
- %XML.Security.Signature

  Argument list: ComputeSha1Digest, CreateX509
- %XML.Writer

  Argument list: CanonicalTree, Caonicalize
- %ZEN.Controller

  Argument list: OnPreHTTP
- %ZEN.Report.Display.node

  Argument list: %GetAbsoluteURL
- %ZEN.Report.reportPage

  Argument list: %DisplayFO, %getFileByRelativeURL, %ToXSLFOStyleSheetLink
- Config.Configuration

  Argument list: AddGlobalMapping, AddNameSpace, GetDataServer, RemoveNamespace
- SYS.Database

Argument list: Compact, SilentIntegrityCheck

2.2.5.15 Class And Class Component Reservations

The following list of classes and class components are being reserved for InterSystems use in the future. Applications making use of them will still function, but they may change in the future without prior notice. Therefore, applications that use these classes or class conmponents should be changed to use alternate, supported pathways to accomplish the same effect. Please contact InterSystems Worldwide Customer Support to discuss the available conversion options.

- %Library.ProcedureContext

  Method: NewResultSet

- %Monitor.Alert

  Method: Create, Delete, GetId

- %Monitor.Manager

  Method: Activate, Alert, AppNotifyMethod, AppSmtpPassword, AppSmtpServer, AppSmtpUserName, Deactivate, Halt, HaltApp, Purge, Refresh, RefreshApp, SignalApp, SmtpPassword, SmtpServer, SmtpUserName, Start, StartApp

2.2.6 Class Compiler Changes

2.2.6.1 Generator Method Ordering

In classes containing method generators, if any generator depends on information about the state of other methods in the class, it needs to indicate this to the class compiler. This is done is including the *GenerateAfter* keyword in the properties of the referring generator method. The keyword is followed by the names of the methods that must be compiled before this generator is run.

2.2.6.2 Class Inheritance Order

There is a new class keyword that determines how to resolve a common member when doing multiple inheritance. This keyword is called *Inheritance* and it can be set to "left" or the default, "right". If a class inherits from two superclasses that have a common member, say a method called **X()**, in classes A and B, for example:

```
Class User.C Extends (A, B)
```

By default, C will get the implementation of **X** from class B, the rightmost superclass takes priority over the other superclasses. This can cause problems if class A has another method that uses **X()** to perform some work and C does not expect to suddenly get an unrelated implementation of **X** from class B instead.

Beginning with this version, class C can specify the inheritance order using the *inheritance* keyword. In this example, the desired behavior is obtained by specifying "left" as the value of the keyword.

2.2.6.3 Remove The cXXXXid and cXXXinheritedid Nodes From ^oddCOM

To increase speed of class compiler and reduce the amount of data it uses, the cXXXXid and cXXXXinheritedid keyword nodes from ^oddCOM. These keywords were internal to the class compiler and should not have been used by any customer code.

However, if an application does need these values, it can call **resolveIds^%occInherit(class,.inheritedid,.id)** to return the values the system used to have.

2.2.6.4 SQL Queries Now Depend on SQLPROC Setting

A query is now projected as a stored procedure only if SQLPROC = 1. Prior to this change, all SQL queries were projected as stored procedures, but those with SQLPROC not set to 1 were hidden from view. Thus these "unmarked" queries were callable but did not appear in catalog queries. Any applications taking advantage of this behavior will need to be updated.

2.2.6.5 Reconcile Use of $ET And Try/Catch In Computed Code

If the class compiler detects that the SQLCOMPUTECODE for a property contains a use of $ET/$ETRAP, then the generated <property>Compute method will not use try/catch. Error processing is therefore the responsibility of the user. This is consistent with a similar change for $ZT/$ZTRAP.

2.2.6.6 Validate Index Names Used In %PurgeIndices

Beginning with version 2009.1, %PurgeIndices validates the name of the indices it is passed. If the index does not originate in the current class then it cannot be purged.

Note: There is an exception to this rule; if the current class is the extent root class and the index is inherited or originates here, it is valid for building and purging.

2.2.6.7 Revise Class Name Normalization For Generator Classes

When compiling an incomplete reference such as **##class(Name).Method()**, the compiler attempts to normalize this classname to a fully-qualified class name. In previous versions, the normalization algorithm used the import list or the origin of this method containing the reference (that is, the class where this method was defined). However, generator methods are effectively implemented in each subclass where they occur.

Beginning with version 2009.1, the compiler normalizes names relative to the current class for generator methods rather than the class where this generator was initially defined.

2.2.6.8 Use Of %this Constrained

In prior releases, applications could access to the private properties of another object by executing code such as

```
New %this
Set %this=otheroref
Set LocalValue=..PrivateProperty
```

Beginning with 2009.1, this code will not work as intended. Applications with similar fragments must be changed. The way Caché enforces public/private protection has been strengthened. Now, a method of class X can access private members of another instance only if the origin of the member is in class X or a superclass of class X.

2.2.6.9 Use Of ##this In Classmethods Prohibited

In prior versions, an instance method could call a classmethod and the classmethod could reference properties within the instance using the ##this reference. Because it is not possible to guarantee a valid value for ##this in classmethods, this usage is now prohibited and will result in an error when the method is executed.

2.2.6.10 Storage Interface Now Checked For Consistency

SERIAL and PERSISTENT classes both use a storage definition to describe how instances of the class will be serialized. The storage definition specifies a type class that will provide the necessary storage interface implementation. Since the requirements of a SERIAL class are different from those of a PERSISTENT class, the class compiler will now check the storage type class for compatibility with the using class to ensure that the proper storage type class is used. An incompatible storage type class will be reported as an error, and the compilation of the class will fail.

2.2.6.11 EXTENT Queries Projected As SQL Procedures

The extent query defined in %Library.Persistent is automatically projected as an SQL stored procedure. If this query is overridden in a subclass of %Library.Persistent, then the subclass determines whether or not the query is projected.

2.2.6.12 %Double.DisplayToLogical Now Returns $DOUBLE Result

This class now always returns a result type of $DOUBLE.

2.2.6.13 INCLUDECLASS Removed

In prior releases, a class could specifiy another class to include into itself – a kind of inheritance. It was never used by customers and has been removed in this release.

2.2.6.14 <Property>Get Now Enforced As Method

In prior releases, it was possible to reference the **<propertyname>Get()** method as if it were itself a property. This means that, for a property named *Age*, all of the following statements compiled and executed:

```
Write someobject.Age, !
Write someobject.AgeGet(), !
Write someobject.AgeGet, !
```

In this release, the compiler syntax checking is improved. The last form is now properly detected and reported as an error.

2.2.6.15 Suppress %%CLASSNAME Generation For Final Extent Root Class

The extent root class is the uppermost class in the hierarchy that instantiates an extent. This means that there are no persistent superclasses that can instantiate instances of themselves. In most cases, the class is one that extends %Library.Persistent (which does not itself allocate storage). The extent root class has the class keyword,*CLASSTYPE*, equal to "PERSISTENT" and *NOEXTENT* "FALSE". Therefore, the list of superclasses of this class that allocate storage, *%%CLASSNAME*, is empty; this is the first such class to do so.

If the extent root class is declared FINAL, then the compiler can be sure that no subclasses exist as well. In this instance, the compiler will not generate the property, *%%CLASSNAME*, since it would always have a null value.

2.2.6.16 Remove IVARmultidimensional Keyword

The planned use for this keyword was never implemented. As part of cleaning up the class compiler, it has been removed.

2.2.7 Object Binding Changes

2.2.7.1 Projections Of %Library Collections

The several "ListOf" and "ArrayOf" collection classes in %Library are now projected externally as follows:

%Library Class	Projection	Superclass
ListOfDataTypes	CacheListOfDataTypes	CacheListOfStrings
ArrayOfDataTypes	CacheArrayOfDataTypes	CacheArrayOfStrings
ListOfObjects	CacheLibListOfObjects	CacheListOfObjects
ArrayOfObjects	CacheLibArrayOfObjects	CacheArrayOfObjects

Each of the classes, CacheListOfDataTypes, CacheArrayOfDataTypes, CacheLibListOfObjects, CacheLibArrayOfObjects has a constructor (equivalent to the %New class method) that takes a CacheConnection argument.

2.2.7.2 Client Name Handling Changes

In prior versions, client names were always deduced from server names by removing or replacing characters not permitted as part of client names in C++ and C#. In 2009.1, if the server defines a client name for some name in the class definition, that name will be used instead of the constructed name. In order to get client names from the server, it is necessary to set the GetClientNamesFromServer property of CacheConnection to true before calling Open() on the connection.

2.2.7.3 MultiValue Collections Projected To .NET

MultiValue collections are now projected to the .NET binding similar to other Caché collections. That is,

- MV.ListOfDataTypes is projected as CacheListOfStrings
- MV.ArrayOfDataTypes is projected as CacheArrayOfStrings
- MV.ListOfObjects is projected as CacheListOfObjects
- MV.ArrayOfObjects is projected as CacheArrayOfObjects

2.2.8 SQL Changes

2.2.8.1 FOR ALL Extension In Caché SQL Deprecated

Caché SQL includes a proprietary extension, FOR ALL. This was added many years ago as a complement to the standard FOR SOME clause. However, in several preceding versions, the use of FOR ALL would cause an SQL parse error. Since none have been reported by customers, it appears that this is not being used in existing application code.

Therefore, in 2009.1 InterSystems deprecates the use of FOR ALL. If any existing application programs use this feature, they should be rewritten since this feature will be removed in a future release.

2.2.8.2 Type Reporting For UNION Queries

The SQL compiler has been changed to report the proper type for columns constructed from UNIONs. In previous versions, a SELECT statement such as

```
SELECT 0 AS var1 FROM Table1
UNION ALL
SELECT 0.1 AS var1 FROM table2
```

would report INTEGER as the column type to xDBC, and the 0.1 value would be sent to the client as an integer with the data truncated.

Now the SQL engine will properly examine all legs of the union and return the type of the highest precedence for each column. The order of precedence for Cache' types is: VARCHAR, DOUBLE, NUMERIC, BIGINT, INTEGER, SMALLINT, TINYINT. So a query such as

```
SELECT MyTinyIntField AS var1 FROM Table1
UNION ALL
SELECT MyIntegerField AS var1 FROM Table2
UNION ALL
SELECT MyNumericField AS var1 FROM Table3
```

will return type NUMERIC for the column since it has a higher precedence than TINYINT and INTEGER. At this time Caché will only use the type precedence in UNIONs as listed above. If other types are involved, no automatic type detection will be done. The type of the union fields with the highest precedence will be returned. Other types will be ignored. To explicitly change the type, use explicit CAST statement.

2.2.8.3 Support INSERT ... SELECT TOP N

Caché SQL now supports the use of the TOP clause in the top-level query of an INSERT ... SELECT statement. For example:

```
INSERT INTO Table2 (field1, field2)
SELECT TOP 10 fld1,fld2 FROM Table1
```

will only insert the top 10 rows retrieved by the select query.

Prior to this change the query would Prepare without error, but the TOP clause would be ignored. This led to confusion where it appeared the results did not match the query. In addition, the TOP clause was not previously supported in subqueries. Now if a SELECT attempts to use the TOP clause in a subquery, an error will be returned when the statement is Prepared. The error is:

```
SQLCODE = -145 :: TOP clause is not valid in a subquery
```

2.2.8.4 New Keyword: %NOTOPOPT

Beginning with Caché 5.2.2, optimizations have been done for TOP ... ORDER BY to decrease the interval before the first row was produced. Usually these changes substantially improved performance. However, on rare occasions, these optimization heuristics make the query worse. For these cases, a %NOTOPOPT hint is now available to force normal total cost optimization processing The syntax for this is:

```
SELECT TOP 5 *
FROM %NOTOPOPT mytable
...
ORDER BY x
```

2.2.8.5 Improve DISTINCT Speed While Preserving Case

Beginning with version 5.2, Caché supported using indexes for DISTINCT and GROUP BY. However, in most cases, this optimization caused case-insensitive columns to be returned in uppercase. To override this behavior, application had to select %EXACT() of the column.

In version 2007.1, Caché allowed a clause to control the behavior of this optimization: FASTDISTINCT ON or OFF. With FASTDISTINCT OFF, the actual row data was returned, rather than a data value possibly in uppercase. Unfortunately, with FASTDISTINCT OFF, an index-assisted DISTINCT improvement occurs if uncollated data is stored with the index. Since most indexes do not store data, and for bit map indexes, the improvement cannot be realized.

2.2.8.6 Correct Processing For FULL JOIN Clauses

Beginning with version 2008.1, Caché supported FULL OUTER JOIN. However, Caché does not support the NATURAL FULL JOIN or FULL JOIN ... USING clauses. These syntax forms were accepted but processed as if they were NATURAL LEFT JOIN or LEFT JOIN ... USING. Beginning in this version, use of these forms will generate an error message rather than incorrectly processing the clauses.

2.2.8.7 Computed Property Changes

Properties that are SQLCOMPUTED can reference the current property value in the SQLCOMPUTE-CODE, if the property is not CALCULATED and it is not a collection.

2.2.8.8 Support UPDATE And DELETE On Abstract Tables

Caché SQL now supports UPDATE and DELETE on an Abstract table where it is assumed the actual row being updated or deleted is in a sub-extent of the table. Prior to this change, SQL considered READONLY and Abstract to be the same resulting in READONLY behavior. Now, if the class is defined as abstract, Caché generates appropriate logic for the UPDATE and DELETE filer to dispatch to the extent of the row being updated for filing.

It is still invalid to attempt to INSERT into an Abstract table; this will result in an SQLCODE=-115 error. In addition, if an application prepares and executes an UPDATE or DELETE statement against an Abstract table that doesn't actually UPDATE or DELETE any rows, a SQLCODE=100 is returned, and not an error.

Note: If an application actually intends to define a READONLY table, the READONLY class parameter must be explicitly declared. A class/table defined as READONLY will return SQLCODE=-115 if an application attempts to prepare an INSERT, UPDATE, or DELETE statement against the table.

2.2.8.9 Perform String Comparisons For %PATTERN, %STARTSWITH And String Operators

When a %PATTERN, %STARTSWITH, "[" (follows), "]" (contains), or LIKE operation is performed in an SQL statement, the operation is done as a String operation. It is assumed both side of the operator are string values and they are treated as such, for example:

```
X %STARTSWITH '5.'
```

or the even more extreme,

```
X %STARTSWITH '-.'
```

If *X* is a Numeric type Caché will not normalize '5.' to a number as with other operators because that changes the intention of the condition (to find all numbers that are 5.<something>, or all negative numbers bigger than -1, respectively.

2.2.8.10 Reduce %Float Usage In SQL

Data declared as %Library.Float is defined as ODBC type DOUBLE. Because of this, when a %Library.Float field used in a WHERE clause and is compared against a literal value or host variable, the value is normalized by calling the appropriate method of %Library.Float. However, if the field uses a subclass of %Library.Float and one of the functions ACOS, ASIN, ATAN, COS, COT, EXP, LOG, LOG10, POWER, SIN, SQRT, TAN, TRUNCATE, or one of the arithmetic operators ("+" , "-" , "*" , "/" , "\" , and "#") are used on the field, Caché can only determine that a numeric value needs to be returned unless the argument to the function is ODBC type DOUBLE, then the function returns DOUBLE.

For example, assume a class named TEST with a property named, ABC and the type of ABC is User.MyFloat, where User.MyFloat extends %Library.Float, and the ODBC type of User.MyFloat is DOUBLE. In the SQL query:

```
SELECT ABC FROM SQLUser.MyFloat WHERE {fn TRUNCATE(ABC,3)}=.482
```

If TEST contains a row where ABC has the value .48259, this row will not be found because the literal input .482 with go through the normalization method of %Library.DOUBLE and $DOUBLE(.482) = .48199999999999998401.

CAUTION: Applications using data of type %Float or extensions of %Library.Float need to be very careful about the actual values of the data. %Library.Float returns a DOUBLE value to ODBC, but does not store a DOUBLE value in the database.

The use of %Library.Float and any extensions of this type is discouraged. The class %Library.Double is preferred instead.

2.2.8.11 Perform Simple Normalization On Input Values

Beginning with 2009.1, SQL statements that expect literal values or host variables as input into the query will now receive those values "lightly normalized" before they are used by the SQL statement. For example, suppose a query has one of the forms

- <field> <operator> <literal>
- <field> <operator> <host variable>
- <scalar function> (<literal>)
- <scalar function> (<host variable>)

in ODBC mode and/or Runtime node. When processing of the query begins, the <literal> or <host variable> will undergo any OdbcToLogical or DisplayToLogical conversion needed (based on the datatype of the item). After this first conversion (if any), the value will then be "lightly normalized" using the following:

- If the value is NUMERIC, INTEGER, or the datatype is %Library.Float, Caché uses the "+" operator any non-null values to normalize the value as a number. This means that a query such as

  ```
  SELECT ... WHERE MyIntegerField = :hVar
  ```

  will return rows where MyIntegerField = 0, if *hvar* has a value like 'abc' (+'abc' equals 0).
- If the value is of type DOUBLE, Caché will convert the value to $DOUBLE (if it is not null).
- If the value is of type TIMESTAMP, Caché will strip any trailing spaces, then if the value ["." it will strip any trailing 0's and then any trailing "."'s. This normalizes '2008-06-27 08:15:23.000' into '2008-06-27 08:15:23', or '2008-06-27 08:15:23.120' into '2008-06-27 08:15:23.12'.

2.2.8.12 Correct Issues With SCALE Reporting And Aggregates

Beginning with 2009.1, a column generated using AVG is reported as type DECIMAL, unless the argument to AVG is a $DOUBLE value; in that instance, the AVG function will return a $DOUBLE value. If the AVG function returns DECIMAL, and the argument to AVG has a precision, *p* and scale, *s*, the precision of the result is 18 and the scale is 18-*p*+*s*. If the AVG function returns a $DOUBLE, the scale for the column will be 0.

2.2.8.13 NEW SQLCODE When Returning Multiple Rows

This version adds a new SQLCODE error, -432, where a function returns multiple rows when only a single value is expected This is currently used in the Informix converted code to report an error when a function is called as a scalar function and is expecting a single value returned, but the procedure returned multiple rows. This corresponds with the Informix error -686.

2.2.8.14 Report Error When Using SUM() on Dates, Times, or Timestamps

There is a new error, SQLCODE=-379, that will be reported when an application attempts to prepare an SQL statement that has a SUM aggregate function with an argument of type DATE, TIME, or TIMESTAMP.

2.2.8.15 An Embedded CALL Statement Creates New %sqlcontext

An embedded SQL CALL statement will now execute the NEW command on the *%sqlcontext* variable during its execution. The *%sqlcontext* variable will be removed when the CALL finishes.

Note: Caché does not support returning result sets from CALL statements in embedded SQL statements.

2.2.8.16 Report Error For INSERT Statements With A Subquery

Caché has never supported a subquery in an INSERT statement such as:

```
INSERT INTO Addressee
  SET LetterId = ? ,
  RelationshipId = ( SELECT id
                FROM Dictionary . AddresseeRelationshipToPatient
                WHERE code = ? ) ,
  AddresseeIEN = ? ,
  recipientType = ? ,
  Name = ?)
```

Prior to this version, it would simply give a syntax error at compile time. Now an error will be returned when the query is prepared/compiled that explains this is not supported. The SQLCODE value for this error -144, “Sub-queries are not allowed in insert statements” .

2.2.8.17 Return BINARY Type For SUBSTRING(<LongVarBinary>)

When using SUBSTRING() with an argument of type LONGVARBINARY, the result returned is now of type BINARY (%Library.Binary), not VARCHAR (%Library.String).

2.2.8.18 Specifying The Collation For A Field In DDL

When a the collation of a field is specified via a DDL CREATE TABLE or ALTER TABLE statement, an error will now be returned if the collation specified is not a collation support by Caché. In previous versions, an indicating an unsupported collation name would be returned during CREATE TABLE, but not during ALTER TABLE. Now, both CREATE TABLE and ALTER TABLE will return a meaningful error message.

Also, when the class is defined and a collation is specified via DDL, the collation value in the collation property parameter will be forced to all uppercase.

2.2.8.19 No Collation Specified Defaults to EXACT

Subscript information for collated fields is now reported as %EXACT when no collation (or so-called EXACT collation) is applied to a property.

2.2.8.20 Return OREFs Instead Of OIDs In SQL Queries

In prior versions, when embedded and dynamic SQL queries returned external streams as part of their results, the references to the streams were returned as OIDs (object ids; %Library.ObjectIdentity). Beginning in this version, the stream references will be returned as OREFs (object references; %Library.ObjectHandle).

WARNING! Existing applications returning external streams as results must be modified to use the new type.

2.2.8.21 Trailing Delimiter Modified After SQL UPDATE

A change has been made to the SQL data filer for UPDATE to prevent a trailing delimiter value from being added to a global node. This happens if %CacheSQLStorage is used with delimited identifiers for the node. For example suppose the node looks like

```
^Patient({ID},10) = {Name} ^ {Age}
```

After an SQL UPDATE, in some cases, the node would be changed to look like

```
^Patient({ID},10) = {Name} ^ {Age} ^
```

While this data is still valid, it may interfere with applications that look at the global data directly. SQL adds the trailing delimiter if the data in the table beyond the trailing delimiter is defined.

2.2.9 SQL Gateway Changes

2.2.9.1 Use %Library.Double For Linked Tables

Caché now represents floating-point values in linked tables by %Library.Double instead of %Library.Float.

2.2.10 CSP Changes

2.2.10.1 Keep CSP Info Off Insecure Channels When Using HTTPS

In previous versions, the CSP sessionId cookie that Caché sent to the browser did not have the 'secure' flag set. This meant that if an https CSP application could force the browser to go to the same web site with http rather than https. In this case, the browser would send the sessionId cookie in clear text which would make it possible to sniff this off the network and then use the cookie information to impersonate this user.

To improve security in version 2009.1, if the CSP session has only ever seen https requests then Caché will send the session cookie with the secure bit set. If the CSP session has seen any http requests, it will the session cookie without the secure bit set. This also means a CSP application that starts on an https page, and then goes to an http page, will start a new session (and take out a license and so on). This is by design as it prevents the sniffing of the sessionId.

If the user application is willing to allow this sessionId to be sent over the network unencrypted there are two ways to accomplish this:

1. Start on an http page first, and then link to the https page. This will ensure the sessionId will never have the 'secure' flag set.
2. When linking to the http page from the https page pass the sessionId with the property CSPCHD=<sessionId>&CSPSHARE=1

2.2.10.2 Convert CSP Parameter Names And Values On Input

In prior versions, when a CSP pages was submitted, Caché did not convert the parameters names based on the characters set of the page. If an application used a non-ASCII name, it would therefore not appear correctly in the %request.Data array. Beginning with this version, Caché will convert both the value and the name.

2.2.10.3 Change Value of Serve FIles For CSP Applications

For the InterSystems-supplied CSP application settings for the csp applications /isc/studio/templates and /isc/studio/usertemplates, the Serve Files parameter has been changed from "No" to "Always and Cached" . In addition, in the portal the default setting for Serve Files has been changed from "No" to "Always" for new applications. This matches the settings in the ^SECURITY routine. Customers may want to review their current CSP application settings, and change the value for their applications to "Always" .

2.2.11 Zen Changes

2.2.11.1 Zen dynaTree Component Improvements

A number of extensions were added to this component:

Display Controls

The component now has the ability to show lines between folder and leaf nodes (like a windows tree control). This is activated by setting the new *showLines* property to true. In this mode, the size and images used by the tree control are fixed. If that is not appropriate, set *showLines* to false.

New Callback Provides Tree Contents

A new callback allows an application to obtain the contents of the tree. This callback is set via the *OnGetTreeInfo* property.

The callback method returns the entire contents of the tree as one array. This structure is designed to make it easy to programatically provide tree contents. Each node in this array corresponds to one node within the tree. At each node there is a $LIST of information about this node:

```
$LB(caption, value, hasChildren, link, expanded, icon)
```

where:

- caption is the displayed text for the node
- value is the logical value of the node
- hasChildren is 1 if there are subnodes, 0 otherwise
- link is an option URL used a link when the node is clicked
- expanded is 1 if the node has children, and they are to be displayed
- icon is the URL of an optional icon to use for this node

The node structure is a directed graph of the form

```
node(0) = $LB(info about this node)
node(0,"ch",1) = "" // index of first child of this node
node(0,"ch",2) = "" // index of second child of this node
```

Support For "Lazy" Loading

If the node structure indicates that a node has children (hasChildren=1), but this children are not in the node graph, then the dynaTree component supports "lazy" loading. The initial contents of the tree are displayed, and when the user expands a folder node that currently has no children loaded, a call is made to server to fetch them. This call invokes the OnGetTreeInfo callback passing it the logical value of the folder node.

Drag Support

This change also adds drag support for the dynaTree.

2.2.11.2 Name Mangling Changes

To simplify bookkeeping for forms, Zen mangles the names of controls used when a form is submitted. CSP reserves control names starting with "Cache" for the purpose of writing login-related pages. With this change, *any* control whose name starts with "Cache" will be considered to be a special case for Login pages and will not have its name mangled.

2.2.11.3 Zen Pages Using SVG

Pages that load SVG dynamically (and do not have an svgFrame in their initial definition) will need to add

```
<page ... useSVG="true">
```

to their page definition. Failure to do so will result in Javascript errors. Zen no longer loads the Javascript code for SVG by default.

2.2.11.4 Changes For radioSet And radioButton Components

Changes were made to the radioSet and radioButton components:

- When disabled, the captions now change style
- A radioSet based on an SQL query, now applies changes to captions correctly
- Captions for radioButton now behave in the same way as radioSet

This change introduces and uses new CSS class for these components to represent the disabled state for captions. These are "a.radioSetCaptionDisabled" and "a.radioButtonCaptionDisabled" . The style for radioButton captions also changes from "span.radioButtonCaption" to "a.radioButtonCaption" to be consistent with radioSet and to support the ":hover" selector.

This means that applications that override the captionClass for this component may have to add addition CSS style information such as

```
color: none;
text-decoration: none;
```

2.2.11.5 dynaGrid Focus Changes

When the edit control within the dynaGrid gets/loses focus, it would raise an *onselectcell* event. In some cases, this lead to extra events being fired. Zen no longer raises this event in this case. Applications that subclass from dynaGrid and depend on this behavior may need to be reworked.

2.2.11.6 Disable Keyboard Shortcuts And Context Keys For Zen Client-Side Menu Widgets

Changes in the keyboard handling in Safari and Firefox 3 effectively disabled the keyboard shortcut mechanism built into the csMenu subsystem. This resulted in the csMenus becoming incompatible with textBox and textArea components on those platforms.

In the short term interest of maintaining some of the utility of the csMenu subsystem, the keyboard handler has been disabled until a cross-platform solution (or bug fixes on the part of Apple and/or Mozilla) are developed. This means that all keyboard shortcuts (including use of the ESC key to close an open menu) have been disabled under Internet Explorer and Chrome as well. Additionally, the csComboBox widget makes heavy use of this same keyboard engine and is likewise disabled.

2.2.12 Zen Reports Changes

2.2.12.1 Allow No Width Or New defaultWidth Attribute On Table

In prior versions, when no width is specified, Zen Reports specified proportional-column-width(1). This can be undesirable in some circumstances when the application actually wants no column-width attribute specified. With this version, if an application specifies width="none", no width attribute specified. This change also defines a new table attribute: defaultWidth, which can be used to over ride the current default width (when no width specified) with whatever the application specifies, even “none” , which means no column-width attribute will be generated.

2.2.12.2 Zen Reports And Table Privileges

In this version, Zen Reports can be used to generate reports (PDF, HTML, or text) even if the application is running as UnknownUser and have no privileges to do anything. This could allow previously invisible information (because of a lack of execute and output privileges) to be seen.

The ZEN Report writer can no longer depend on lack of privileges to execute an operating system command to protect information. Instead, sensitive information in tables must be protected through user security or ability to run CSP applications with appropriate safeguards. Users are expected to protect the tables/result sets/stored procedures from which ZEN Reports derives its information with SQL privileges. If an application has permission to view the table, then it can generate XML from it and produce a report.

2.2.12.3 Element Handling Redefined

The specification for element generation is now defined as the following: for each row of a ResultSet that a group processes, an element will display once in the group. Applications that expect only one element may now fail; there is now one element per row of ResultSet processed by a group even if an element repeats. It is always possible to NOT repeat elements by using "SELECT DISTINCT" or other mechanisms in the generation of the result set.

2.2.12.4 Individual Margin Settings Overridden By Non-Null Margin Setting

In this version, the values for *marginTop*, *marginBottom*, *marginLeft*, and *marginRight* are ignored if the *margin* setting is non-null.

2.2.12.5 Custom Aggregate Definitions Changed

Custom aggregates **Var** and **StDev** now correspond to MDX in being unbiased estimators, that is, dividing by n-1 where n is population size. The new, biased estimators (divide by n) are now **VarP** and **StDevP**.

2.2.13 Web Services Changes

2.2.13.1 SOAP Client Wizard — Multiple Return Values

A method that returns multiple values is now represented as explicitly having Output parameters. This change is required to handle the case where an optional parameter is missing. In prior versions, this would cause the first value returned to be incorrectly assigned as the return value of the method.

2.2.14 xDBC Changes

2.2.14.1 Privilege Checking Extended To More Catalog Queries

When calling the following catalog queries, privilege checking is now enforced and no catalog metadata will be returned if the user does not have at least one privilege for the table or view.

ODBC	JDBC
SQLSpecialColumns (SQL_BEST_ROWID=1)	getBestRowIdentifier()
SQLForeignKeys (Cross Reference)	getCrossReference()
SQLForeignKeys (Exported Keys)	getExportedKeys()
SQLForeignKeys (Imported Keys)	getImportedKeys()
SQLPrimaryKeys	getIndexInfo()
SQLStatistics	getPrimaryKeys()

2.2.14.2 Improvements To Precision/Scale For Numeric Values And Expressions

The following assumptions are now made when preparing SQL statements via xDBC:

Numeric literals

When a numeric literal value is specified in an SQL statement that is Prepared through xDBC, its value is replaced with a host variable (unless it is enclosed in ((val))). The type of this variable is NUMERIC with a length of 20, a precision of 18, and a scale of 9. If a different type, precision, or scale is desired, a CAST should be used in the SQL statement.

Addition and Subtraction

The type of the resulting expression will be NUMERIC unless one or both of the arguments is a double, then the result will be a DOUBLE. If the type is NUMERIC,

- the precision will be determined by:

max(scale1, scale2) + max(precision1-scale1, precision2-scale2) + 1

- and, the scale is determined by:

  max(scale1, scale2)

Multiplication

The type of the result will be NUMERIC unless one or both of the arguments is a double, then the result will be a DOUBLE. If the type is NUMERIC,

- the precision will be determined by:

  min(18, precision1 + precision2 + 1)

- and, the scale is determined by:

  min(17, scale1 + scale2)

Division

The type of the result will be NUMERIC unless one or both of the arguments is a double, then the result will be a DOUBLE. If the type is NUMERIC,

- the precision will be determined by:

  min(18, precision1 - scale1 + scale2 + max(6, scale1 + precision2 + 1))

- and, the scale is determined by:

  min(17, max(6, scale1 + precision2 + 1)))

2.2.14.3 Return ODBC 3.5 Catalog Metadata

Beginning with 2009.1, ODBC catalog queries have been updated to support ODBC 3.5. This has the following implications. (The column name changes do not affect backward compatibility because applications bind by column number.)

MetaData Table Name	ODBC 2.0	ODBC 3.x
SQLSpecialColumns	PRECISION	COLUMN_SIZE
SQLSpecialColumns	LENGTH	BUFFER_LENGTH
SQLSpecialColumns	SCALE	DECIMAL_DIGITS
SQLTables	TABLE_QUALIFIER	TABLE_CAT
SQLTables	TABLE_OWNER	TABLE_SCHEM
SQLColumns	TABLE_QUALIFIER	TABLE_CAT
SQLColumns	TABLE_OWNER	TABLE_SCHEME

MetaData Table Name	ODBC 2.0	ODBC 3.x
SQLColumns	PRECISION	COLUMN_SIZE
SQLColumns	LENGTH	BUFFER_LENGTH
SQLColumns	SCALE	DECIMAL_DIGITS
SQLColumns	RADIX	NUM_PREC_RADIX
SQLColumns	<none>	COLUMN_DEF
SQLColumns	<none>	SQL_DATA_TYPE
SQLColumns	<none>	SQL_DATETIME_SUB
SQLColumns	<none>	CHAR_OCTET_LENGTH
SQLColumns	<none>	ORDINAL_POSITION
SQLColumns	<none>	IS_NULLABLE
SQLColumnPrivileges	TABLE_QUALIFIER	TABLE_CAT
SQLColumnPrivileges	TABLE_OWNER	TABLE_SCHEM
SQLForeignKeys	PKTABLE_QUALIFIER	PKTABLE_CAT
SQLForeignKeys	PKTABLE_OWNER	PKTABLE_SCHEM
SQLForeignKeys	FKTABLE_QUALIFIER	FKTABLE_CAT
SQLForeignKeys	FKTABLE_OWNER	FKTABLE_SCHEM
SQLForeignKeys	<none>	DEFERRABILITY
SQLProcedureColumns	PROCEDURE_QUALIFIER	PROCEDURE_CAT
SQLProcedureColumns	PROCEDURE_OWNER	PROCEDURE_SCHEME
SQLProcedureColumns	PRECISION	COLUMN_SIZE
SQLProcedureColumns	LENGTH	BUFFER_LENGTH
SQLProcedureColumns	SCALE	DECIMAL_DIGITS
SQLProcedureColumns	RADIX	NUM_PREC_RADIX
SQLProcedureColumns	<none>	COLUMN_DEF
SQLProcedureColumns	<none>	SQL_DATA_TYPE
SQLProcedureColumns	<none>	SQL_DATETIME_SUB
SQLProcedureColumns	<none>	CHAR_OCTET_LENGTH

MetaData Table Name	ODBC 2.0	ODBC 3.x
SQLProcedureColumns	<none>	ORDINAL_POSITION
SQLProcedureColumns	<none>	IS_NULLABLE
SQLPrimaryKeys	TABLE_QUALIFIER	TABLE_CAT
SQLPrimaryKeys	TABLE_OWNER	TABLE_SCHEM
SQLProcedures	PROCEDURE_QUALIFIER	PROCEDURE_CAT
SQLProcedures	PROCEDURE_OWNER	PROCEDURE_SCHEM
SQLStatistics	TABLE_QUALIFIER	TABLE_CAT
SQLStatistics	TABLE_OWNER	TABLE_SCHEM
SQLStatistics	SEQ_IN_INDEX	ORDINAL_POSITION
SQLStatistics	COLLATION	ASC_OR_DESC
SQLTablePrivilges	TABLE_QUALIFIER	TABLE_CAT
SQLTablePrivilges	TABLE_OWNER	TABLE_SCHEM
SQLGetTypeInfo	PRECISION	COLUMN_SIZE
SQLGetTypeInfo	MONEY	FIXED_PREC_SCALE
SQLGetTypeInfo	AUTO_INCREMENT	AUTO_UNIQUE_VALUE
SQLGetTypeInfo	<none>	SQL_DATA_TYPE
SQLGetTypeInfo	<none>	SQL_DATETIME_SUB
SQLGetTypeInfo	<none>	NUM_PREC_RADIX
SQLGetTypeInfo	<none>	INTERVAL_PRECISION

2.2.14.4 Add IS_AUTOINCREMENT Column To getColumns Catalog Query

The JDBC DatabaseMetaData method getColumns() now returns a twenty-third column titled, IS_AUTOINCREMENT. This column will return YES if the column is one that is automatically incremented upon INSERT, otherwise NO is returned.

2.2.14.5 Change To Query Return Value

This version of Caché now always returns a value for **%SQLGatewayConnection.Fetch()**. However, it no longer returns the value 100 when all the data has been consumed. Applications must examine the value of SQLCODE to determine this.

2.2.14.6 Preserve Leading Zeros In Fractional Seconds Of SQL_TIMESTAMP

The fractions portion of the SQL_TIMESTAMP structure is an integer that represents the number of nanoseconds. Prior to this release, there was a conversion problem in the ODBC driver when there were leading "0" s before additional digits. For example,

```
2008-02-02 12:23:45.0012
```

would be interpreted as if it were

```
2008-02-02 12:23:45.12
```

This error is corrected in version 2009.1.

2.2.14.7 JDK1.4 Deprecated

JDK 1.4 is deprecated as of this release. It is officially out of support by Sun (END OF SERVICE LIFE) except through extended contracts that require payment. JDK 1.4 developed libraries (JAR files) can run without issue in most cases on JDK 1.5+. JDK 1.4 support will dropped as of Caché 2010.x.

The current InterSystems plans for Java are to continue all new product development on JDK 1.6. JDK 1.5 will be supported for current features (and subsequent bugfixes). As of the policy above, JDK 1.5 will reach the end of its service life on 30 October 2009. InterSystems will continue to support for JDK 1.5 for a period of at least 1 year after this date.

Generally, InterSystems will continue to support each version of the JDK for at least one year after its end of service date. This will allow product features to evolve and take advantage of virtual machine improvements in stability and speed.

2.2.15 MultiValue Changes

2.2.15.1 MVBasic MATWRITE And MATREAD Statements And Triggers

The MATREAD and MATWRITE statements will now call the appropriate trigger if one is defined for the file. In prior versions, the trigger was ignored.

2.2.15.2 SYSTEM(11) Changes

SYSTEM(11) will now return a boolean flag for whether the default select list exists in these emulations: Pick, Information, PIOpen, IN2, and Universe. Previously it returned a count of the number of items in the list. Applications relying on the previous (incorrect) implementation will have to change to a different method of getting the list count.

2.2.15.3 Change Line Numbers Reported By SYSTEM(49)

In prior versions of Caché, the line number reported by SYSTEM(49) was relative to the intermediate (MVI) routine instead of the source (MVB) routine. It will now be adjusted to properly show the source line number. Also, a <SUBSCRIPT> error would be thrown if the routine had been recompiled while it was running. SYSTEM(49) will now report the line number as 0 if the actual source line can not be determined.

2.2.15.4 Sort WHERE Command By Port Number

The WHERE command outout is now sorted by port number by default. This makes Caché consistent with other commands such as LISTU and WHO. In addition, there is a new (D) option to WHERE which allows the sort and selection to be based by the Caché job number. When the (D) option is used , then the selection criteria is Caché job number, not MV port number.

2.2.15.5 Correct Echoing And Prompts On MultiValue INPUT And IN Statements

Beginning in version 2009.1, Caché obeys the following rules for the **INPUT** , **INPUT @** , **IN** and **KEYIN** functions relating to how they echo, what device they echo to, and what prompts they support (the PROMPT statement).

1. If any echoing is to be done, it is only done to the screen, never to the printer.
2. The prompt string, defined by the **PROMPT** statement, is always honoured for the **INPUT** and **INPUT @** statement. The prompt string only goes to the screen, never to the printer.
3. The **INPUT @** statement, when echo is turned off, needs to be emulation dependent. Some platform emulations will echo the characters during **INPUT @** even when echo turned off (for example, Cache, Universe), while others (such as jBASE , and D3) need to respect the echo status.
4. The **KEYIN** and **IN** statements, when echo is turned on, are emulation-dependent. Some emulations will honour the echo setting and others will never echo the character.

2.2.15.6 Remove Pagination From Some Emulations For MultiValue

Beginning with version 2009.1, some MultiValue emulations will no longer paginate their output by default. In those emulations, applications wishing to have paginated output must explicitly enable it with the **SP.CONDUCT** command. Now, only Cache, Universe and Unidata emulations paginate output by default.

2.2.15.7 New Functions Added

The functions LISTVALID and LISTNEXT have been added to MVBasic. The semantics of each is the same as the ObjectScript functions $LISTVALID and $LISTNEXT, respectively. Applications that use either of these identifiers as variables will need to be edited to remove the name conflict.

2.2.15.8 Open Trigger No Longer Invoked

Because an error in a trigger routine could prevent trigger commands from being run, trigger commands no longer call the OPEN trigger.

2.2.15.9 Truncate Fractional Dates To Integer For OCONV

Prior to this version, an OCONV of fractional date values would sometimes not do any conversion, instead returning the original value. Now fractional values will always be truncated to an integer before conversion.

2.2.15.10 Match Select List To Emulation Correctly

The PROC P command now correctly retains or throw away the active select list, depending on the emulation mode, as follows:

- In Cache, UniVerse, PICK, Prime, IN2, PIOPEN, and Unidata emulations, the active select list is retained, after both MVBASIC programs, TCL commands, and queries.
- In the other emulations, such as jBASE, D3, R83, and REALITY (among others), the active select list is not preserved after running an MVBASIC program.

2.2.15.11 READNEXT Correction To Handle An Empty List Variable

In prior versions, READNEXT from an empty list variable would read from the default list 0 instead of treating it as an error. Beginning with this release, READNEXT will now check for "" and treat that as an invalid OREF, setting STATUS() = -2.

2.2.15.12 Changes To INDEX Search Of Empty String

Using CACHE emulation in previous versions, the expression

```
INDEX(string, "" ,1)
```

would return 0. In this version, it now returns the correct value, 1.

2.2.15.13 MAT READ Gives Error For Non-MultiValue Array

MATREAD will now throw an error when an application attempts to read from a non-MultiValue input. In previous versions, MATREAD would execute the ELSE clause. MATREAD has also been changed to provide a more descriptive error code when a non-MultiValue array is used in an operation that requires an MultiValue array.

2.2.15.14 READ Setting Changes

The default setting of READV0 for Ultimate emulation was changed from RV0.KEY to RV0.EXISTS. The default setting of READ.RETAIN was changed for these emulations – Information: off; PIOpen: off; Ultimate=: on.

2.2.15.15 SPOOLER Heading Processing Changed

In this version, a number of changes have been made to the HEADING and FOOTING statements, and general printer/terminal output in this context. Applications that depend on the format of SPOOLed output should be carefully checked to make sure the output still conforms to expectations.

2.2.15.16 Reset Spooler Job Numbers Daily

In previous versions, the job numbers for MultiValue spooler print jobs was never reset. Thus, after some weeks or months the values become very large and difficult to manage. In this version, the job number will be reset to 1 every day. Checks will be made to ensure existing jobs are not overwritten, for example if job 1 exists already then we'll use job number 2, and if that exists then we'll use job number 3 and so on. But application can no longer predict spooler job numbers.

2.2.15.17 Ignore SQLTABLENAME And SQLPROJECTION If MVASSOCIATION Present

SQLPROJECTION defines the default projection of a collection to SQL. For MVENABLED classes,

```
SQLPROJECTION = TABLE
```

indicates that a child table is to be projected as well as a column. The SQLTABLENAME property defines the name of that table.

MVASSOCIATION allows properties in an MVENABLED class to define multiple collections as forming one SQL child table. If MVASSOCIATION is defined for a property, then SQLPROJECTION and SQLTABLENAME are now ignored and the value of MVASSOCIATION is used as the name of the child table projected by the associated properties.

This change alters the name of SQL tables projected by MVASSOCIATION. The prior behavior was wrong and not expected by customers using MVASSOCIATION. Some MultiValue customers may have to adapt their applications to recognize the new (and correct) table name.

2.2.16 Jalapeño Changes

2.2.16.1 NetBeans 5.5 Plugin Removed

Jalapeño now only builds for the NetBeans plugin for version 6.0; not both 5.5 and 6.0.

2.2.16.2 Select Access To Compiled Class Table Required

Jalapeño now examines all known subclasses of a given class to determine which instance to use. It does this by using the %Dictionary.CompiledClass table to locate them. Therefore, a user needs SELECT privilege on this table for the operation to succeed.

2.2.16.3 Exception Generated For Non-Unique Client Name

It is possible that several different Caché classes refer to the same Java class using clientname. In this case, the Jalapeño Object Manager is unable to determine which of those Caché classes should be used. In previous versions, it used the first one returned by a query. Now it checks that only one class satisfies the condition, and returns an error if this is not the case.

2.2.16.4 Object Save Algorithm Changed

This version changes the way Jalapeño saves multiple objects in a single method call. It moves most of the logic of resolving dependencies between objects from the Caché server to the Java client. The new implementation of save should be functionally equivalent to the old one but since more logic is now moved to Java client changes in the behavior (for example, timing changes) may manifest themselves.

In particular, before this change if an object being saved had a collection, the type of this collection was always preserved. In the new algorithm, to optimize the save, this might be no longer true because an ArrayList can be replaced with a LazyList.

2.2.16.5 Schema Builder Handles Arrays Differently

This version changes the behavior of the schema builder for some Java properties that are primitive arrays. Suppose in Java we have an array such as one of the following:

```
byte [] SomeBytes;
char [] SomeChars;
int [] WholeNumbers;
```

By default such properties are projected as a Caché Collection which is not usually a convenient choice. What is worse, if they are annotated like:

```
@CacheProperty(type = "%CacheString")
```

they were projected as List of %CacheString. Now, the following rules determine how a primitive array is projected:

1. As collection of primitive types if it is not annotated. This is the same as previous versions.
2. As single property with a given type if it is annotated with

   @CacheProperty(type = "...")

   and is NOT annotated as either @Collection or @Lob. This differs from previous versions.

3. As a collection of given types if annotated as: @Collection(type = CollectionType.LIST)@CacheProperty(type = "...") (no change)

   @Collection(type = CollectionType.LIST)@CacheProperty(type = "...")

   This is the same as previous versions.

4. As %Stream if annotated as

   @Lob (type in @CacheProperty ignored)

   This is the same as previous versions.

2.2.17 Platform-specific Items

This section holds items of interest to users of specific platforms.

2.2.17.1 Change Default Telnet Translation From Raw To UTF8 On Unicode Systems

The following locales had their default translation for Telnet (“Other Terminal”) changed from RAW to UTF8:

- araw
- csyw
- danw, deuw
- ellw, engw, enuw, espw, eurw
- finw, fraw
- hebw, hunw
- itaw
- ltuw
- nldw
- plkw, ptbw
- rusw
- thaw

When this default was created, almost no telnet emulator supported UTF-8; it made sense to use RAW. Currently, most emulators support UTF-8 by default. Furthermore, those locales not based on Latin1 (for example, araw, csyw, ellw, hebw, hunw, ltuw, plkw, rusw, and thaw) had problems with non-ASCII characters because these characters fell outside the range supported by RAW and would thus

be translated as "?". Finally, the Latin1-based locales in the above list (such as deuw, enuw, ptbw, and so on) were not able to handle the Euro sign via telnet using RAW because this character value is also above 255.

UTF-8 is fully compatible with ASCII. This means that locales with languages that use ASCII exclusively (such as English) will not notice any change. Customers with other Latin1-based locales which use characters between 128 and 255 might need to change the encoding used by their telnet client programs to UTF-8 so that they can correctly handle accented letters.

2.2.17.2 Terminal Log Files From Unicode Systems Now In Unicode

Beginning with version 2009.1, the log files produced by the Caché TERMINAL will be encoded as ANSI for 8–bit installations, and as Unicode for Unicode installations.

2.2.17.3 $LISTSAME On Big-Endian Systems

On prior versions, the **$LISTSAME()** function could report that two lists were not equal when one list contained an integer and the corresponding position in the other list was a different representation of that same integer (such as string or floating point). The failure occurred only on big-endian platforms.

2.2.17.4 MQ Context Changed

Prior versions of Caché unconditionally set the Context to 2 (SET_ALL_CONTEXT). This behavior caused authorization errors when using MQ.

Now, a flag is defined for an MQ Put connection (Net.MQSend) called Context. The context flag is:

- 0: Default.
- 1: Identity. The application should use the Identity context to set and pass the ApplIdentityData field.
- 2: All context. The Context allows the ApplIdentifyData and the PutApplType fields to be set and passed.

Setting the context to Identity (1) or All (2) may have authorization implications. Applications that succeeded before may now fail with an MQ Authorization error (2035). Please see your IBM Websphere MQ documentation for additional details.

2.2.17.5 OpenVMS Stream Record Size Increased

Caché now allows applications running on OpenVMS to create records on OpenVMS in Stream mode up to 32767 bytes in length (the maximum allowed by OpenVMS RMS sequential files). This length includes the CR/LF that ends the record. Prior versions of Caché would handle requests to write records longer 163853 by silently splitting them into 2 (or more) records in the sequential file. This change will also report a <WRITE> error if the 32767 limit is exceeded; applications will get a clear indicator that they have exceeded the maximum record size, rather than unknowingly creating multiple records.

2.2.17.6 Comparisons On 64–Bit Systems Corrected

Caché version 2008.2 introduced a discrepancy in floating-point comparisons between nearly equal values that is corrected in this version. The problem occurred in comparisons between a $DOUBLE value and a Cache decimal value where the 20 digit expansion of the $DOUBLE value (rounded to be a decimal value) was equal to the Cache decimal value. In this case, comparing the $DOUBLE value with the Cache decimal value resulted in all three of the relations less than, equal to, and greater than being false. Now, these comparisons always have at least one of the relationships as true.

2.2.17.7 Windows Interfaces Now Built With Visual Studio 2008

Version 2009.1 now builds its Windows components using Microsoft Visual Studio 2008. Applications using C, C++, CallIn, or CallOut should also be built using that version of Visual Studio. The Windows samples in the SAMPLES namespace were also built with that version.

2.2.17.8 Windows Installation Now Uses Lowercase Directory Names

Caché now uses lowercase for installation directory names. In previous versions, for example, it installed into the \Bin directory. In version 2009.1, it is \bin.

2.3 Operators

2.3.1 System Management Portal

There have been a number of changes and improvements to the system management portal since 2007.1 was released. These are detailed in the administrator section.

3

Caché 2008.2 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2008.2 that, because of their difference in this version, affect the administration, operation, or development activities of existing 2008.1 systems.

General upgrade issues are mentioned at the start of this document. Those customers upgrading their applications from releases earlier than 2008.1 are strongly urged to read the upgrade checklist for earlier versions first. This document addresses only the differences between 2008.1 and 2008.2.

3.1 Administrators

This section contains information of interest to those who are familiar with administering prior versions of Caché and wish to learn what is new or different in this area for version 2008.2. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

3.1.1 Journal Restore UI Changed

The journal restore utility, **^JRNRESTO**, asks a sequence of questions of the operator to determine what part of the journal to restore. The questions and their order are different in this release from previous releases.

CAUTION: Those sites that use a script to supply the answers to questions presented by **^JRNRESTO** must be reviewed to ensure they still accomplish their intended task.

3.1.2 Prevent Login As UnknownUser

This verison of Caché no longer allows "UnknownUser" as the user name for login. The UnknownUser is now reserved for use to identify users who gain access as unauthenticated users. Any application using "UnknownUser" as a login name will need to be changed to use some other userid.

3.1.3 Support Variable-Sized Routine Buffers

In prior versions, Caché allowed the administrator to specify in the CPF file the amount of memory to be used for routine buffers (*routines*) and the size of each buffer (*rbufsiz*). Assuming sufficient memory, Caché allocated a single pool of fixed size buffers; each buffer holding one executing routine.

In this version of Caché, *rbufsiz* is ignored in the configuration file. Caché instead allocates half the space specified by *routines* to a pool of 64KB buffers, three-eighths of the space for a pool of 16KB buffers, and one-eighth of the space for a pool of 4KB buffers.

Note: The maximum number of buffers allocated to any pool is limited to 65,529. Caché also will never allocate less than 205 buffers to any sized pool. This means the actual memory used for routine buffers can be larger than specified in the configuration file.

Important: The format for Caché routines does not allow more than 32K characters for literal strings regardless of the setting for the maximum routine size.

3.1.4 Superserver Support For Secure Communications

The Caché superserver has been enhanced to allow communications using either the Secure Sockets Layer (SSL, version 3) or Transport Layer Security (TLS, version 1). Individual ports can be configured via the Management Portal to have SSL required, enabled or disabled. For each setting, the following action takes place:

- Required: If the initial attempt to connect does not present a valid SSL or TLS certificate, the superserver will drop the connection.
- Enabled: If the initial attempt to connect presents a valid SSL or TLS certificate, the superserver will mark the connection as secure. Otherwise, the connection will be marked as unsecure.
- Disabled: If the initial attempt to connect presents a valid SSL or TLS certificate, the superserver will drop the connection.

Configuration of the ports with respect to communications security is done via the Management Portal: **[Home] > [Configuration] > [Shadow Server Settings]**. Both the "Add Shadow Server" and "Edit Shadow Server" pages now have an Advanced options, SSL Configuration.

A Caution Regarding Requiring SSL on the Superserver for 2008.2

If an administrator creates an SSL configuration called %SuperServer, enables this configuration, and then specifies in the Superserver SSL/TLS Support field that the SSL/TLS is required (clicking the Required radio button), then the System Management Portal will no longer be able to communicate with the instance.

The administrator can correct this problem at the Terminal (which still connects) using ^SECURITY as follows:

1. Start ^SECURITY in a terminal session.
2. Select option 9 (Systems parameter setup).
3. Select option 1 (Edit system options).
4. In answer to the question, "What type of SSL/TLS connections are allowed for the super server?," change the setting to Accept or None (equivalent, respectively, to Enabled and Disabled in the Management Portal).

3.1.5 System Scan On New Install or Upgrade

Beginning with this version of Caché, upon completion of a successful install or upgrade, Caché will automatically scan the related files and directories compiling an inventory of the instance. This data will be stored in the database, %SYS.

3.1.6 National Language Support

The administration of NLS has been extensively reworked in this version of Caché. A new page has been added to the Management Portal to provide a convenient means of administration and the %SYS.NLS class now contains the functionality of the previous NLS utilities.

The program previously used for NLS management on Windows, "cnls.exe" , has been removed and replaced by the line-oriented routine, **^NLS**. The routines ^NLSMGR, ^%Wsnls, ^%Wsnls2, and LOCGEN have also been removed and new system classes, %SYS.NLS.Table, %SYS.NLS.Format, %SYS.NLS.Locale and %SYS.NLS.Device, have been added.

^NLS is a line-oriented utility that provides basically the same functionality of CNLS.EXE in previous releases. Its main options are:

1. Locale definitions
2. Table definitions
3. Current system settings

The first two options manipulate static definitions of the main NLS entities: locales and tables. The third option allows the user to see currently active settings on a system-wide basis or just for the current process.

3.1.7 Security-Related Changes

3.1.7.1 New Resource Required For Secure Login

A new resource is now created during installation called, **%Service_Login**. This resource is now used by the **$SYSTEM.Security.Login()** function. The **$SYSTEM.Security.Login** function will now only succeed if the application calling that function runs as a user who has **USE** access to this resource.

Before this change, any application could call **$SYSTEM.Security.Login()**, and if presenting a valid userid and password, then login as that userid. Now, they must also be able to **USE** the resource.

Note: This resource is not required if the calling process has **Write** permission on the resource that protects the CACHESYS database, or the calling routine is stored in the CACHESYS database.

3.1.7.2 Adding Roles Prohibited From Command Line

The function, **$SYSTEM.Security.AddRoles()** can no longer be called from the command line, or from within the debugger. Doing so could allow a user to inadvertently leave their process with elevated roles when the call returns. Calling the method in this way will fail to elevate the roles, and the error

```
ERROR #940: Insufficient privilege for operation
```

will be returned as the value of the function. Moreover, if

- there is a routine which calls **$SYSTEM.Security.AddRoles()**, and
- it encounters anunhandled COS error in this routine while in programmer mode, and
- the routine breaks at the debug prompt,

the programmer can no longer call **$SYSTEM.Security.AddRoles()** from the debug prompt and escalate roles outside of the scope of the application.

Note: It is recommended that any routine or class use error handling in the procedures which call **$SYSTEM.Security.AddRoles()**.

3.1.7.3 Privileges For Changing Database Properties Modified

To edit the properties of a database, the privilege **%Admin_Secure:USE** is now required. If a user without this privilege attempts this action, the following error will occur:

```
ERROR #921: Operation requires %Admin_Secure:USE privilege
```

3.1.8 Parameter File Name Always Defaults to cache.cpf

The default for the optional configuration argument used when starting Caché from the command line has changed. Previously, when a command line such as

```
ccontrol start an_instance_name a_config_file
```

was issued to start Caché, a_config_file.cpf would become the default configuration file used to start later instances.

Beginning with this version, the configuration file will be used only for the current startup. Subsequent start commands which do not explicitly specify a configuration file will cause the Caché instance to be started using the cache.cpf configuration file.

3.1.9 Platform-Specific Items

This appendix holds items of interest to users of specific platforms.

3.1.9.1 UNIX: Improved Installation

The "cinstall" package now offers three setup types to replace "Standard" and "Custom".

- Development: (the default) install server and all client components.
- Server: Install the components for Caché server; but do not install client components, nor anything normally found in the /dev area.
- Custom: Install server components; present a dialog for all client components, allowing them to be individually selected.

Furthermore, the install will use a parameters.isc file created from a previous installation to parameterize the install rather than by prompting the user.

Note: Automatic install scripts may need to be reworked because the order of some of the Caché installation prompts has changed.

3.1.9.2 UNIX: Protection Against Starting Multiple Copies of Caché

In prior versions, the use of .lck files for Caché databases to control failover proved insufficient to prevent multiple instances of Caché from being started against the same databases under all circumstances. Caché now takes out additional locks for the control process, write daemon and journal daemon to provide better isolation.

As part of this change, information has been removed from the cache.ids file, and is placed in a system-internal file. The existence of the cache.ids is all that is used as a marker.

3.1.9.3 Macintosh: CShell Effective Id Anomaly

The setregid call used by csh to establish the real and effective group id for executing commands ignores the effective group id (presumably as a security issue). Administrators should use alternatives such as sh, ksh, or bash when administering Caché from other accounts than the one Caché runs as.

This issue appears on all Macintosh supported platforms.

3.1.9.4 Windows: License Acceptance Dialog

The Windows installer now requires explicit acceptance of the InterSystems license terms as the first step in the installation process. This applies to both upgrade and new instance installations. The **Next** button for the license acceptance dialog is not enabled until user selects the **I accept the license terms** choice. The only other action available is to cancel the installation.

3.1.9.5 Windows 64–Bit: Studio Activate Wizard Unavailable

The Activate Wizard is currently supported only on 32–bit windows platforms. It is greyed out on the Studio **Tools** menu for 64–bit systems.

3.2 Developers

This section contains information of interest to those who have designed, developed and maintained applications running on prior versions of Caché.

The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

3.2.1 Compatibility Notices

3.2.1.1 Objectscript Compiler Version

The objectscript compiler has been modified to improve code generation. This change is significant and internally the minor version of the compiler has been incremented to mark this. Code for routines or methods compiled in this version of Caché will not run on previous versions.

3.2.1.2 Class Component Signatures

This version of Caché corrects a deficiency in the class compiler in versions 2007.1 and 2008.1. Existing classes that compiled without error in those releases may now be recognized as having signature errors in some overridden inherited components.

3.2.1.3 Property References

This version of Caché corrects a deficiency in the class compiler that existed in earlier versions. References to properties must have an instance context; you cannot GET a property without giving the OREF for it.

This change also affects cases where an applications called propertyGet() methods as class methods. For example, applications using ##class(name).property to call the propertyGet() will have to change to call a proper class method. Such calls will now result in an error. These method calls should be rewritten as class methods instead of calculated property.

3.2.1.4 Preservation Of $TEST Value Across Calls

It is documented that the value of $TEST is preserved across calls to any procedure. In prior versions, however, this was true only for calls to procedures defined in the same routine as the caller. Caché has been changed in this version so that the value of $TEST is restored regardless of where the called procedure is defined.

3.2.1.5 Change To Storage Global Name Generator

The name of the global used to hold the storage for instances of a class is derived from the fully-qualified name of the class; that is, package name and class name. If the length of the resulting name exceeds the maximum global name length, the class compiler may truncate either the package name, class name or both to generate a global name that fits within the defined limits.

When one of the name parts is truncated, the truncated portion is encrypted to help create a unique name. In previous versions, this encryption produced a three-character string. Beginning with this version of Caché, the algorithm produces a four-character string.

CAUTION: If an application has external class definitions that do not include storage definitions, it is possible that recompilation of the class will produce a storage structure incompatible with existing data.

3.2.1.6 $SYSTEM.Task To Be Removed

Caché version 5.2 moved the task manager classes from %SYSTEM.Task to %SYS.Task. The documentation for the class were changed to indicate that the %SYSTEM.Task class was deprecated. Users were advised to change their applications to use %SYS.Task.

Note: InterSystems intends to remove the %SYSTEM.Task and the corresponding $SYSTEM.Task classes in the next version of the product.

3.2.2 Character Set Changes

3.2.2.1 Support For Unicode-16 And JIS2004 Added

Caché introduced support for Unicode in version 3.0. In this first implementation, only characters in the Basic Multilingual Plane (BMP), that is, those in the range U+0000 to U+FFFF, were supported. Version 2007.1 introduced support for GB18030 in one- and two-byte values.

This version extends that support to codepoints that result in Unicode surrogate pairs. These are pairs of 16-bit characters that, taken as a unit, represent Unicode characters outside the BMP. A surrogate pair consists of a high-order and a low-order code. The high word ranges from U+D800 to U+DBFF and the low word from U+DC00 to U+DFFF. When interpreted as a UTF-16 character, these pairs map from U+10000 to U+10FFFF and cover one million code points. Unicode strings that contain surrogate pairs are said to be encoded in UTF-16.

Existing Objectscript string functions such as $Extract, $Find, and so on are not aware of surrogate pairs and thus can provide inconsistent results when supplied with strings that may contain such pairs. This version of Caché introduces specialized versions of the regular string functions that correctly handle surrogate pairs. These functions are:

- $WAscii
- $WChar
- $WExtract
- $WFind
- $WIswide
- $WLength
- $WReverse

These new functions must scan the string in order to determine the boundaries between the characters because some may be represented by a single 16-bit value and others by surrogate pairs. For example, the 4 word sequence 0x41, 0xD800, 0xDC00, 0x42 is a 3-character string if surrogate pairs are taken into account because the two middle words correspond to a single character, U+10000.

Note: The new W functions work as expected when supplied with regular 16-bit Unicode characters but are generally slower than their non-W versions due to the overhead of having to scan the target strings. Therefore, one should preferably use the existing non-W functions and resort to the new W ones only when there is a real possibility that the strings will contain surrogate pairs.

On 8-bit systems the $W* functions work as their non-W equivalents. In this case $WIswide() always returns zero.

3.2.2.2 New Collation Defaults For Some Unicode Locales

Some Unicode locales have had their default collations replaced with a new collation that allows any Unicode character to be encoded in a subscript. Their previous default collations were restricted to the 256 characters in their 8-bit base character set.

Locale	Prior Default Collation	New Default Collation
danw	Danish1	Danish2
deuw	German2	German3
ellw	Greek1	Greek2
espw	Spanish1	Spanish2
finw	Finnish2	Finnish3
plkw	Polish2	Polish3

The new collations were available in prior versions, but they were not the default. The older collations will remain available.

Note: The new default is compatible with the old one in that alphabetic characters collate the same way. However, punctuation and control characters may collate differently since they are ordered as they are in Caché Standard.

3.2.2.3 Default For Simplified Chinese Changed

The Simplified Chinese locale (chsw) now uses GB18030 as the translation default where GB was previously used (Magtape, Printer, Caché Terminal, Sequential files, $ZCONVERT). The GB table has been removed from the list of available tables.

3.2.3 Routine Changes

3.2.3.1 ^NLSMGR, ^%Wsnls, ^%Wsnls2, and LOCGEN

These routines have been deleted as a result of the re-implementation of national language support.

3.2.3.2 Permissions For %ERN and %SS

We now allow a user with the **%Development:USE** permission to run %ERN and %SS. Previously we would only allow someone with **%Admin_Operate:USE** permission to run it. Write access to CACHESYS is still required to run %SS.

3.2.4 Objectscript Changes

3.2.4.1 Explicit Conversion To $DOUBLE Now Required

Programs that depend on the automatic conversion of large decimals numbers to IEEE double precision numbers will now require an explicit call to **$DOUBLE()** to make that conversion.

Whenever a numeric literal is larger than 9223372036854775807E127 (or smaller than -9223372036854775808E127), a <MAXNUMBER> error will be generated. The error will also be generated whenever an arithmetic calculation involving decimal numbers exceeds this range. This reverts to the Caché behavior before version 5.2.

In version 5.2, IEEE double precision numbers were introduced. In that release, IEEE double precision numbers were generated by the **$DOUBLE** function and by decimal operations that exceeded the range for these numbers. Starting with version 2008.2, only the **$DOUBLE** function can be used to initially create IEEE double precision values. Out-of-range decimal literals and decimal calculations will generate the <MAXNUMBER> error. Some undefined computations involving decimal functions (example: $ZLOG(0)) will now return the <ILLEGAL VALUE> error rather than returning the IEEE infinity or NaN value.

Any operation with operands that are a mix of decimal numbers and IEEE double precision numbers will continue to have an IEEE double precision result. The newly defined **$DECIMAL** function will be available to convert IEEE double precision numbers to decimal numbers.

The reasoning behind this change is this. The mixing of decimal floating point with binary floating point caused some problems since their introduction. The Caché decimal type has almost 19 digits of accuracy; the IEEE binary floating-point type has less than 16 digits of accuracy. The IEEE binary double precison floating-point type has a much greater range (about 1.7976931348623158E308) than the Cache' decimal type (9223372036854775807E127 or 9.22337203685477580 7E145). Thus, a conversion in either direction will lose either range or precision. Also, most decimal fractions do not have an exact representation in binary floating point. Therefore, most conversions of fractional decimal numbers to IEEE binary floating-point numbers will involve a small change in value.

To reduce the impact of these incompatibilities between decimal floating-point values and binary floating-point values, InterSystems has decided to not automatically convert between these values. A conversion directly between these numeric values will require a direct call on the **$DOUBLE** or **$DECIMAL** functions.

3.2.4.2 New $ISVALIDDOUBLE Function Added

This version of Caché adds new function calls **$ISVALIDDOUBLE(num)** and **$ISVALIDDOUBLE(num,scale,min,max)**. These functions are identical to **$ISVALIDNUM** except they always test validity for the $DOUBLE type.

The value of min, and max are always converted to the IEEE 64-bit binary floating-point type. If the num argument has a valid format, it is convert to the IEEE 64-bit binary, floating-point type. The min/max range is tested using binary floating point comparisons. If min is not supplied then **$DOUBLE("-INF")** is used. If max is not supplied then **$DOUBLE("INF")** is used. The value **$DOUBLE("NAN")** is always considered valid regardless of the values of min/max.

3.2.4.3 New $DECIMAL Function Added

This version of Caché implements **$DECIMAL(X, N)** which converts the IEEE double-precision value X to a decimal string with N significant digits. Rounding is done according to the IEEE Floating Point standard. Currently, values of N larger than 40 are unsupported.

If N has the value 0, then the result is a string with 20 significant digits and with special rounding. The result of **$DECIMAL(X, 0)** is also the default conversion of the IEEE double-precision value X to a string.

When $decimal(X,0) is evaluated the following rounding is done:

- No rounding is needed if the result can be represented exactly in 20 or fewer significant digits.
- If the result has more than 20 significant digits, and if the 20th decimal digit is a 0 or a 5, then it is rounded up to a 1 or a 6. All other digits in the 20th position will be left unchanged. Digits beyond the 20th position will be discarded or replaced with zeros as is necessary to put the string into numeric canonical form.

Note: If a value with this special rounding is later converted to 19 or fewer decimal digits, then this second conversion will be the correctly rounded result of the original value, without a double-rounding error.

Also, the string value that results from **$DECIMAL(X, 0)** will collate in the correct order with the string values that result from converting a Cache decimal number to a string.

3.2.4.4 $ISVALIDNUM Changes

The function **$ISVALIDNUM()** has been changed. In this version of Caché, **$ISVALIDNUM** will return 1 only for syntactically correct strings. All others will return 0.

3.2.4.5 $DOUBLE Changes

The **$DOUBLE()** function now accepts "INF" and "NAN" as arguments resulting in the IEEE values infinity and not-a-number, respectively.

In keeping with normal Caché practice, strings beginning with "INF" and "NAN" produce the proper value even if these strings are followed by extraneous material. Thus, **$DOUBLE("INFRACTION")** produces a positive infinity, and **$DOUBLE("NANTUCKET")** a NaN. Also, an optional number of arithmetic signs may proceed the string value as in **$DOUBLE("-+-inf")**. The case of the string is not relevant.

3.2.4.6 Infinity and NaN

Note: A new draft standard for IEEE binary floating-point is under development. Under this standard, the valid strings for **$DOUBLE** will be restricted to "INF", "INFINITY", "NAN", and "SNAN" (a signalling NaN is not distinguished by Caché from a non-signalling NaN). Developers are urged to restrict themselves to these proposed strings with a single optional sign character.

3.2.4.7 Controlling Overflow On Conversions

This version of Caché now allows the developer to control the behavior of programs in the presence of numeric conversion errors. This is done via the function, $ZUTIL(68,70,x).

- If the value of "x" is 1, when **$DOUBLE** or **$ZDCHAR** encounter a numeric string with an absolute value too large to be a finite value of the IEEE 64-bit binary floating-point type, a <MAXNUMBER> error is reported.
- If the value of "x" is 0, no error will be reported. Caché will replace the value with the appropriate value of infinity, "INF" or "-INF".

For example, executing:

```
Write $ZUTIL(68,70,0)
Write $DOUBLE("1e309")
```

does not result in a <MAXNUMBER> error; instead the string "INF" is written as the value.

Note: The default value for overflow control when a process starts is 1.

3.2.4.8 Conversion Of Numeric Literals

During compilation when parsing a numeric literal, or at runtime when converting a string to a numeric value, Caché will convert the numeric value to a $DOUBLE (IEEE 64-bit binary floating-point type) if the value exceeds the range of the Caché decimal floating-point type. Note that this extends the numeric range, but it also reduces the numeric precision by about 3 decimal digits.

In this is not the desired behavior, the function, $ZUTIL(68,45), can be used to have these conversions from numeric string to decimal return the most-positive/most-negative decimal floating-point value when the range of decimal floating point is exceeded. Also, the function, $ZUTIL(68,70), which can be used to control error reporting when the range of $DOUBLE (IEEE 64-bit binary floating point) is exceeded.

3.2.4.9 Exponentiation to the Zero Power

In prior versions, Caché defined zero raised to the zero power as zero. This was true regardless of the numeric representation. Thus, *X****Y* always evaluated to zero when X and Y were both zero, regardless of representation – 0, $DECIMAL()), or $DOUBLE(0).

This interpretation is at odds with the proposed IEEE Floating-Point Standard which defines any IEEE value (including NaNs and infinity) raised to the zero power as 1. Accordingly, this version of Caché changes the definition of $DOUBLE(0) raised to the zero power to conform to the standard.

3.2.4.10 $FACTOR Improvements

In prior versions of Caché, calls to **$FACTOR** on negative numbers, INF and NAN did not give consistent results. This has been corrected by the following changes:

- **$FACTOR** no longer computes approximate results and it no longer limits the number of significant bits to 64. **$FACTOR** now computes all the bits in the integer part of its arguments and returns the appropriate bit string.
- When **$FACTOR** is applied to a negative argument, it returns the bit string corresponding to the absolute value of that argument. When **$FACTOR** is applied to **$DOUBLE("NAN")** or **$DOUBLE("INF")**, it returns an empty bit string.

3.2.4.11 $LISTSAME Changes

The operation of the function, **$LISTSAME**, has changed slightly. When comparing items from each list argument:

- elements with the value $DOUBLE(+0), $DOUBLE(-0) and 0 are treated as being identical.
- element with the value $DOUBLE("NAN") to be the same as any other list element containing $DOUBLE("NAN").

Note: Caché does not distinguish between different NAN representations even though the hardware underlying different platforms may generate different representations. Also, some NANs are considered to be signaling and others are considered to be quiet. Caché behavior does not depend on these differences.

3.2.4.12 THROW Without Argument Deprecated

This version of Caché deprecates the use of **THROW** without an argument. In its place InterSystems strongly recommends explicitly passing the exception as an argument to **THROW**. To understand the reasoning, consider the code fragment:

```
TRY {
    Some statements
}
CATCH {
    Error correction statements
    …
    THROW
}
```

If the execution of "Some statements" raises an error, the code for the **CATCH** block is executed. If "Error correction statements" do not themselves cause an error, the **THROW** will cause the error that occurred when "Some statements" were executed to be passed to an enclosing error handler.

If, however, the execution of "Error correction statements" causes an error, then the error from "Some statements" will be overidden and the enclosing error handler will receive this latter error information. The error that occurred in the **TRY** block will not be reported.

InterSystems recommends that the code be rewritten as

```
TRY {
    Some statements
}
CATCH ErrorData{
    Error correction statements
    …
    THROW ErrorData
}
```

This ensures that, if the **THROW** is reached, the error information will be passed to the enclosing error handler. If the execution of "Error correction statements" causes a failure, this will be reported to the enclosing error handler, but the exception data from the **TRY** block will still be available for debugging.

In the rewritten example, if the error is detected by Caché, ErrorData will be an instance of %Exception.SystemException. This class is reserved for use by Caché. Applications that wish to provide their own exception data should define subclasses of %Exception.AbstractException and use instances of these subclasses as arguments for the **THROW**.

Note: The main distinction between system exceptions and user exceptions is how they are processed. Instances of %Exception.SystemException will be processed by the nearest enclosing **CATCH**, **$ZTRAP**, or **$ETRAP** statement.

User-defined subclasses of %Exception.AbstractException will only be processed by enclosing **CATCH** statements. If there is no enclosing **CATCH**, a <NOCATCH> error will be thrown.

Thus, the reason for deprecating the use of **THROW** without an argument is the difference in the way system-generated versus application-generated exceptions are handled. The use of an argument allows precise control over the kind of exception process executed.

3.2.4.13 Supported Macros

This version of Caché defines the initial set of officially supported ObjectScript macros. These macros will be the only ones displayed by StudioAssist when the sequence, "$$$" , is recognized during program input. The full set of macros and their definition can be displayed via the $SYSTEM.OBJ.ShowMacros method.

3.2.4.14 Changes To Compiled Code

This version of Caché eliminates the tokens that were previously generated to mark the beginning of each line source line. The system now keeps track of the number of CacheObjectScript (or CacheBasic, CacheMVBasic, or CacheMV PQ/PQN PROC) statements that have been executed in a manner more efficient than previously.

This gives a much more accurate picture of how much work is going on in a routine. In previous versions, blank lines, comment lines, and lines with executable statements on them were all marked with these tokens. This change also frees programmers from worrying about the performance effects of writing in a much more readable, modern style, with each statement on a separate line, and with ample comments.

3.2.4.15 JOB Command Requires Startup Directory

Among the "process-params" given to the JOB command is *os-directory*, a directory the child job should switch to as part of starting up. This is not necessarily the directory the job will use as it may be overridden by the default directory associated with the namespace for the child job.

In prior versions, errors that occurred in the child job while attempting to switch to the *os-directory* were not treated consistently across platforms. On Windows and OpenVMS, errors were ignored. On UNIX, the behavior varied by platform.

All platforms have now been changed to return a <DIRECTORY> error to the original job command if the child fails to switch to the specified o/s directory.

3.2.5 Class Changes

3.2.5.1 Class Compiler Correction — Signatures

In Caché 2007.1 and 2008.1, there is a deficiency in the class compiler mechanism for inheritance. This affected all inherited components except methods; methods are processed correctly. Specifically, the compiler failed to properly reconcile the attributes of the inherited component with the values possibly overriding them in the subclass.

This deficiency has been corrected in this version and those occasions where a subclass improperly overrides attributes of an inherited component will now be reported as signature errors in the component.

3.2.5.2 Class Compiler Correction — Properties

Prior to this version of Caché, it was possible to attempt a property "Get" on a class without supplying an instance context. However, Caché does not permit properties to be associated with classes similar to Java class constants; properties are associated only with class instances. The generated code for this case in prior versions was unpredictable as to what it would return.

This version of the class compiler reports an error that the result of the attempted operation is undefined.

3.2.5.3 Class Compiler — Journaling

When the journal switch was changed from being set individually for globals to being a database property, one effect was that class compiles were then journaled. However, if a class compile fails, the database it applies to will be in an inconsistent state until it is successfully recompiled. This means that there is no need to journal the class compile itself.

In this version, journaling of class compiles is off by default. It can be enabled using the qualifier '/journal=1'.

Note: Systems using journaling to maintain a shadow system, relying on a compile on the main system being picked up on the shadow, should change the default qualifiers so that updates made as part of the compile will be reflected on the shadow system.

3.2.5.4 Class Compiler — Query Compilation

In prior versions of Caché, all queries defined in a class were callable as stored procedures, even though only the ones marked as SQLPROC appeared in catalog queries. In this version, a query that is not explicitly defined as SQLPROC is no longer callable as a stored procedure. Any applications taking advantage of the older behavior will need to be updated.

3.2.5.5 Class Compiler — Keeping Query Intermediate Code

Beginning with this version, Caché no longer generates MAC code for the generated table routines; it now generates .INT routines and these are retained only if the "k" flag or /keepsource qualifier is specified. This behavior now matches that of the generated class routines.

3.2.5.6 NLS Changed To Use Object Interface

The **^%nls** routines have been rewritten for this release using Caché object technology. Applications that used **^%nls** directly in prior releases must be modified to use one of the methods found in the classes %SYS.NLS or %Collate.

3.2.5.7 New Argument Added to $SYSTEM.SQL.TuneTable

A new (sixth) optional argument has been added to the method, **$SYSTEM.SQL.TuneTable**. This argument expects a boolean value indicating whether to clear the SELECTIVITY and EXTENTSIZE values from the specified table.

If this argument is true, and the second argument (which indicates whether to update the table values) is also true, the SELECTIVITY and EXTENTSIZE will be reset.

Note: If the class is deployed, the class definition will not be updated.

3.2.5.8 $SYSTEM.OBJ.ShowMacros() Method added

The **ShowMacros()** method has been added to the $SYSTEM.OBJ class in this version of Caché. Invoking the method displays the documentation and definition of the supported Objectscript macros.

3.2.5.9 $SYSTEM.OBJ.GetClassList(.() Method added

The **GetClassList()** method has been added to the $SYSTEM.OBJ class in this version of Caché. Invoking the method returns all classes in the current namespace as a local array subscripted by the class name. Class selection can be controlled by the following qualifiers, each of which is assigned a value of 0 or 1:

- /application=<N> - Include application classes (<*N*> = 1)
- /system=<N> - Include system classes, that is, ones with class attribute 'system' set to something other than zero (<*N*> = 1)
- percent=<N> - Include percent classes (<*N*> = 1)
- /mapped=<N> - Include classes mapped from other databases (<*N*> = 1). or just classes in default databases passed to the method (<*N*> = 0)

More information on qualifiers is displayed by the method call, **$SYSTEM.OBJ.ShowQualifiers()**.

3.2.5.10 %SYNC.Transporter Class Added For Moving Objects

The %SYNC.Transporter is new in this version of Caché and is intended for copying persistent objects from one namespace to another. The class provides facilities for both sending and receiving objects. On the sending side,

- An instance, T, of the Transporter class is created.
- The **Import** of T is called to get the objects to be copied. The import operation automatically creates new objects or updates existing objects as required.
- When all objects in this exchange have been registered, the **ExportFile** method is called to save the registered objects in a system file.

On the receiving side,

- An instance, T, of the Transporter class is created.
- The OIDs of the objects to be sent are registered with T.

3.2.5.11 %Studio.SourceControl.File Class Added

This version of Caché adds a new class %Studio.SourceControl.File that just imports/exports classes, routines, and so on to a file in the filesystem. The item is determined to be editable if the user has write permission on the file.

This is an elementary source control class that can easily be enhanced to actually talk to a source control system. It is intended to be used as a base class for customer source control classes, or as a code example.

3.2.5.12 %Library.File Default Directory

In the class, %Library.File, the following methods will use the value of $ZUTIL(168) (the current default directory), if no value is specified for the directory argument:

- GetDirectory()
- GetDirectoryLength()
- GetDirectoryPiece()
- GetDirectorySpace()

3.2.5.13 %Stream.FileBinary And %Stream.FileCharacter Classes Added

This version of Caché introduces two new classes that correspond to %Stream.GlobalBinary and %Stream.GlobalCharacter. These use the new stream formats but store their data in files rather than globals.

The directory where the data is stored can be specified at the property level, for example,

```
property FileStream As %Stream.FileBinary(LOCATION="c:\temp");
```

The older classes, %Library.FileBinaryStream and %Library.FileCharacterStream, are unchanged; they will continue to work as before. However, if writing a new class it is suggested you use these new file stream classes.

3.2.5.14 Class %Library.EnumString Added

A new system datatype class has been implemented in this version of Caché, %Library.EnumString. This class is identical to %Library.String except that the string values are restricted to those specified in the *VALUELIST* parameter of the class:

- The **LogicalToOdbc** method will look up the value of the string in the *VALUELIST* parameter and display the corresponding value in *DISPLAYLIST*.
- The **OdbcToLogical** method will do the opposite; it looks up the value in *DISPLAYLIST* and returns the corresponding value from *VALUELIST*.

with the addition that VALUELIST->DISPLAY list transformation is done for LogicalToOdbc conversion, and DISPLAYLIST->VALUELIST transformation is done for OdbcToLogical conversion.

3.2.5.15 Class %SYS.Date.SlidingWindow Added

This class defines the commonly used entry points of the **%DATE** utility. The %SYS.Date.SlidingWindow class supports entry points for inspecting, setting and modifying the system-wide or process-specific sliding window definitions. Each class method with the exception of **GetSystem()** and **GetProcess()** returns a status indicating success or failure.

The %SYS.Date.SlidingWindow class implements almost all entry points of the **%DATE** utility and with this change makes this mechanism available to all Caché scripting languages. The functions of **%DATE** are implemented with slightly different names in %SYS.Date.SlidingWindow to make them more readable.

Note: The %DATE utility is now deprecated. It will be removed from Caché distributions in some future version. Existing applications should convert to using the methods of %SYS.Date.SlidingWindow.

The following functions of **%DATE** are not implemented in %SYS.Date.SlidingWindow and will no longer be available once **%DATE** is removed from Caché distributions: **CvtAbsStart**, **CvtAbsEnd**, **CvtRelStart**, **CvtRelEnd** and **LEAP**.

3.2.5.16 Navigating %Library.RelationshipObject

Several methods have been added to the %Library.RelationshipObject class to make it similar to a collection object. These methods can be called on a relationship whose cardinality is either MANY or CHILDREN. The following methods are now implemented:

- GetObjectIdAt()
- GetObjectIdNext()
- GetObjectIdPrevious()

Note: It is important to not that while %Library.RelationshipObject implements a collection-like interface, it is *not* a collection class and is not required to implement the entire collection interface. Relationships are different from collections even though they implement many similar behaviors.

3.2.5.17 Inventory Scan Task Added To %SYS.Task

A new on–demand task has been added to the task manager. The "Inventory Scan" task (%SYS.Task.InventoryScan) runs the **Inventory.Scanner.RunScan()** method and saves the resulting inventory scan.

Inventory scans start from the instance install directory and perform a recursive examination of all directories, files, databases and routines within databases, recording their name, size, date last modified and a hash identifier. This data is stored into the %SYS database.

The Inventory Scan task is set to run automatically after the install or upgrade finishes. Users may start it manually thereafter, if they make changes to the system.

3.2.5.18 TSQL Implies PROCEDUREBLOCK

Starting with this version, defining a method whose language is set as TSQL automatically assumes PROCEDUREBLOCK equal to 1 overriding any explicit setting for the class or method.

3.2.5.19 CDL Qualifiers Removed

Object flags and qualifiers ("4" and "version4compatible" , respectively) referring to the ability to export data in CDL have been removed.

3.2.5.20 FetchODBC Method Removed From Default Query Interface

Prior to this release, the **FetchODBC** method was introduced to provide an interface to send query data to an xDBC client. Now, Caché has added a **SendODBC** method that much more efficiently sends the query data to the xDBC client and removes the need for the **FetchODBC** method. Therefore, **FetchODBC** code that just calls the **Fetch** query method and reformats the data is no longer generated.

If a user had written a **FetchODBC** method, the **SendODBC** method will still user the users code as it would do before.

Note: The **FetchODBC** method is unlikely to have been used by user code as it was previously only used by Caché to send data to xDBC. If an existring appl;ication relies on this **FetchODBC** method being generated, it will need to change to call the **Fetch** method instead.

3.2.5.21 Wider Recognition of LogicalToStorage and StorageToLogical Methods

The **LogicalToStorage** and **StorageToLogical** methods convert logical values to and from storage values respectively. Prior to this version, only methods inherited from the property type class (or property class) were recognized. This limitation was due to the way SQL recognized property methods.

In this version, inherited methods as well as locally overridden methods are recognized. It is now possible to implement **LogicalToStorage** and **StorageToLogical** methods for a property, and both SQL and Object actions will invoke them at the appropriate times.

3.2.5.22 %IO.FileStream Changes

The functionality of the method, **Clear()**, as been augmented by the methods, **TruncateAt()** and **ExternalByteTruncateAt()**. In addition, methods **OutputToDevice()** and **CopyFrom()** have been improved to do more accurate global timeout handling when copying from a sluggish stream.

3.2.5.23 Preserve Stream Content Across Restarts

Previously, when an application created a file stream where the LOCATION was not specified, Caché created it in the system wide temp stream location. If the application then saves the file stream, Caché marked it as permanent in the database, but did not move the stream from the directory.

The system wide temp stream location was automatically cleaned up on a Caché restart, so all such file streams were being deleted.

To address this issue, Caché now does the following:

1. Caché will now allow a default stream directory configuration parameter for each namespace. If not specified, it will be assumed to be the stream subdirectory of the location of the CACHE.DAT file.
2. When a stream file property is first initialized, if there is no LOCATION property parameter, Caché sets the directory to the default stream directory for this namespace.
3. If an application writes to a stream where the directory has never been set, it will create a temporary file in the system wide temp directory. Saving this stream will move the file from the temp directory to the default stream directory for this namespace.

This will minimize change to existing stream behavior while ensuring that we do not lose any file. The current behavior of streams is:

1. An application that opens an existing file stream or links to an existing file and then writes to the stream will create a new temporary file in the same directory as the existing file. If this is saved then the existing file is deleted and the temp file renamed so it has the same name as the linked file.
2. An application that links to a filename that does not exist, and writes to the stream,will create a new file with the filename specified. Saving this stream marks it as permanent.
3. For a class that defines a file stream as a property of a persistent object, creating a new instance and writing to trhe stream will create a new file in the default stream directory. Saving this object will mark the stream as permanent.
4. An application that creates a new file stream object outside of a property, and does not specify a directory in the %New command, will create a new temporary file in the system wide temp directory when the stream is written. When the stream is saved, it will be moved to the namespace specific default stream directory.

3.2.5.24 Handling Of Time On Import Has Changed

The way Caché handles the last-modified time specified in files imported by XML and %apiRTN changes in this release to make it more consistent. The current rules are:

- For classes, Caché allows the TimeChanged class attribute to be exported during XML export of a class definition. This may be suppressed by the */diffexport* qualifier.
- On import, if there is no last-modified time in the file, Caché will use the following algorithm:
  - If the class already exists and is identical to the previous copy, Caché uses the previous copy *timechanged* value.
  - If the class already exists and the *timechanged* is the same as the file time modified, then Caché uses the current time. This is because it must make sure this node is modified since the class is different from the previous one.
  - Otherwise, Caché sets the *timechanged* to the file time modified.
- When an routine or class is imported, if the last-modified in the file differs from the last-modified time of the file, Caché assumes that the file has been modified by some external tool, but the timestamp in the file has not been updated; it uses the last-modified time of the file. This ensures the timestamp of the item in Caché changes so it can detect that this item has changed.

Note: This change has also been made to time-modified handling by %R code.

3.2.5.25 %Monitor.Alert.External() Method Added

The %Monitor.Alert.External() method has been added to provide way for user applications to send alerts via SNMP or WMI. The method takes parameters that describe the alert and uses a generic Cache

SNMP Trap (cacheAppAlert) or the WMI Cache_Event to send the information. See the %Monitor.Alert class documentation for more details.

3.2.5.26 File Archiving Classes

This version of Caché adds classes that permits connection to an external document retention service. The classes allow for the definition of content, %Archive.Content, as well as the management of the initiation of the data transfer, %Archive.Session. In this first release, the archive interface has been tested for use with the EMC Centera™ servers.

3.2.5.27 Row ID Flag Added to ResultSet Metadata

In the column metadata returned for queries and result sets, bit 12 is used to indicate that the column is part of the RowId..

3.2.5.28 %BuildIndices and %PurgeIndices Now Validate Names

The methods%BuildIndices and %PurgeIndices accept a $LIST of index names to use for their operation. Starting with this version, that list is now validated. If it contains a name that is not the name of an index that can be built in the current class, then the method returns an error status.

3.2.5.29 FreezeOnError Removed From System Classes

The FreezeOnError property for a Database is now always set to on. Therefore, all the FreezeOnError references have been removed from the system classes that define, manipulate or display this property. Specifically,

- SYS.Database: the FreezeOnError property has been removed. It is also no longer returned by the **Detail** query.
- SYS.Metrics: the FreezeOnError property has been removed from the database metrics display.

It has also been removed from the Management Portal.

3.2.5.30 XMLNew Method Argument Changes

The first argument is changed from type, “tree” , which is the index of the DOM in the global to type, “document” , which is a %XML.Document instance.

The second argument is renamed from node to nodeId for clarity.

There is now an optional third argument to **XMLNew** which is the oref of the already instantiated containing object. This allows seeing the object context when overriding **XMLNew**. The containerOref argument is passed by **XMLImport** when importing referenced objects as the containing object. The containerOref argument is “” when passed by %XML.Reader for Correlate'd objects as there is no instantiated containing object in this case.

See the documentation for the class, %XML.Adaptor, for further details.

3.2.5.31 Translation List For Portuguese Updated

Uppercase accented characters of the Portuguese language were added to the character set of the %Text.Portuguese class. This allows Portuguese text containing uppercase accented characters to match with unaccented variations of the same word. The complete list of accented characters for the Portuguese language that are treated in this fashion is: “ÀÁÂÃÇÉÊÍÓÕÔÚÜ” .

3.2.5.32 TCP Broadcast Logic Changed

In prior releases, when the target process of a broadcast using the **$zu(94)** function has a TCP device as $Principal, the output data was buffered for the device. It was not transmitted until either the buffer became full or the process issues a subsequent "write !" to the device. In this version of Caché, the broadcast message will be transmitted immediately.

3.2.6 MultiValue Changes

3.2.6.1 Account Emulation Setting Affects MVBasic Command Line

Before this change the command was compiled with the options set by the most recent BASIC routine compile, or by entering a $OPTIONS setting on the command line.

Starting with this version, when a BASIC command is entered at the command prompt on the terminal, it will now be compiled using the account emulation settings. If a non-default option is required (one which is not the CEMU default), the $OPTIONS and the command must both be entered as one command line, separated by a semi-colon.

3.2.6.2 Debugging Display

While debugging an MVBasic routine at a terminal, the <BREAK> message will now include a line showing the name of the MVBasic source routine and the source line number.

Note: Many lines of executed code may map to the same source line if that line is a $INCLUDE statement.

3.2.6.3 MVIMPORT For Universe Backups

This version of Caché changes the way Universe backups are imported, namely:

- The O option has been removed.
- If an account already exists, then MVIMPORT will not import the backup over it. The user must import to a non-existent account.

- If MVIMPORT can locate a namespace of the same name as the account would use by default, then MVIMPORT will restore to that namespace if it is currently empty.

Tip: InterSystems recommend that users specify the "W" option when importing from Universe so that MVIMPORT will display its plan, then pause and ask for confirmation.

To have fine control over the location of databases, the "W" option will ask the user to confirm the accounts on the backup and the namespaces that MVIMPORT would use for them. Then quit out of MVIMPORT at the prompt and use the SMP to create the databases and namespaces in the desired locations. Rerun MVIMPORT and, if the plan is correct, continue with the restore.

3.2.6.4 MVIMPORT Change To Contents Of IMPORTED.VOC

In prior versions, the file, IMPORTED.VOC, contained the MVIMPORT exceptions, that is, all the items that were NOT written to the VOC. It now contains all the original VOC, regardless of whether or not the items were restored to the VOC. This allows a customer to see what the VOC looked like on their original system before MVIMPORT made decisions on whether or not to apply the items to the VOC.

3.2.6.5 Reference To Undefined Variables Returns <UNDEFINED> Error By Default

MVBasic routines will now be subject to the switch that controls all references to undefined variables in Caché. By default, MVBasic routines that attempt to reference an undefined variable will now get an <UNDEFINED> error. Before this change, an empty string was silently substituted for the undefined variable.

To change the default behavior:

- For the current process –

  The process can invoke the either $ZUTIL(18, 2) or $ZUTIL(68, 72, 1) to restore the previous behavior.

- For the entire system –

  Invoke $ZU(69, 72, 1) during system startup, such as in a ZSTU routine. Any new processes started after that point will have the previous behavior.

3.2.6.6 $GET Function Added

The function, $GET, has been added to MVBasic. This is a MultiValue analog of the Caché function, $GET, except that it does not have the Caché abbreviation of $G.

Note: Prior to this change, *$GET* was a valid variable name. This is no longer allowed. Applications using *$GET* as a variable will have to be changed to use a different name.

3.2.6.7 Collation Changed

In prior versions, right justified fields that contain zero or null would sort into the same place in an index. Starting with this version, nulll fields will sort before fields containing zeros.

CAUTION: Any indexes on right justified fields should be deleted and re-built.

3.2.6.8 ED Changes

It is now possible to enter value marks or sub-value marks in ED the same way as other MV platforms. CTRL+\ will enter a sub-value mark **$CHAR(252)** and CTRL+] will enter a value mark **$CHAR(253)**.

The TB command now works inside ED. For example,

```
TB 3,6,9
```

means set tab stops at 3, 6 and 9. If the user enters a tab at any point in the ED editor, the tab gets replaced with a number of spaces to satisfy the TB setting.

The delimiter between tab stops can be spaces or commas. The TB setting persists until the user logs off and on again.

3.2.6.9 TCL Changes

Caché now supports the TABS command. This is a TCL command and the format and usage is identical to the TB in the ED editor, for example:

```
USER: TABS 3,6, 9
```

3.2.6.10 #PRAGMA Added

By default, when an MVBasic routine is compiled, Caché generates a routine name MVB.xxx. This name applies to the intermediate MVI source code and to the object code. If you want to specify a specific routine name for a source file, add a #PRAGMA statement to the file:

```
#PRAGMA ROUTINENAME=rrr
```

where "rrr" is the routine name and where "rrr" satisfies the syntax for Cache routines: first character is % or alphabetic, and subsequent characters are alphabetic, digit or period. If the name supplied is not in this format, an "invalid routine name" compiler error results.

Once a routine name has been associated with a source file, that name will continue to be used even if the #PRAGMA is removed. To stop using the associated name and have the system generate a new name, specify an empty routine name:

```
#PRAGMA ROUTINENAME=
```

Tip: The empty name should be specified only once (for one compile). Then the #PRAGMA should be removed. Otherwise the system will continue to generate a new name each time the routine is compiled, instead of replacing the existing routine.

3.2.6.11 SYSTEM(31) Now Returns A Unique Value

SYSTEM(31) has been changed; it now returns a unique id for each Universe-related emulation. Prior to this, it did not distinguish among these.

3.2.6.12 SYSTEM(30) and SYSTEM(40) Now Return Port Ids

The **SYSTEM()** function which reports the owner of a lock after a failed lock operation will now return the port number instead of the process id of the owner. For D3 emulation, it is SYSTEM(30); for all other emulations, it is SYSTEM(43).

The process id continues to be available in STATUS() after the failed operation.

3.2.6.13 SYSTEM(0) In D3 Emulation Changed

In prior versions, for D3 emulation, SYSTEM(0) returned the lock owner. Now it will return the current STATUS() value, which will vary depending on the last operation that set a status value.

3.2.6.14 $CHAIN Support Removed

This directive was originally introduced in PICK to compensate for restrictions on the size of programs. In Caché, this issue does not arise. Applications attempting to use this directive will receive a syntax error and must be upgraded to use a more modern program linking mechanism, for example, subroutine calls.

3.2.6.15 Handling Breaks On INPUT

The **Ctrl-C** (break) sequence, if entered while executing the INPUT statement, will now go to the COS debugger. In prior versions, the application would return to the command line.

Note: This assumes the application has not disabled this by executing BREAK OFF.

3.2.6.16 Log File Collects Erroneous Usage

When a MultiValue program uses a non-numeric string in a numeric operation, a value of zero is used. When this happens, Caché now writes an entry in the mv.log file when this action is taken., Developers can review this file and take action to correct the program or data. The file, mv.log, is an operating system file in the Caché installation Mgr directory. The entry will indicate the program line that used the incorrect value.

3.2.6.17 Support For MVBASE Dimmed Foreground Colors Added

The MVBASE emulation now supports the sequences @(-57) through @(-64) to produce the dimmed foreground colors white, yellow, magenta, red, cyan, green, blue, and black, respectively.

3.2.6.18 Terminal Type Additions

New terminal types vt100-color, vt220-color and vt330-color have been added in this version. They were added rather than modifying existing definitions to conform to common practice of creating "-color" definitions in cases where color schemes are optional.

3.2.6.19 Support For XPICKCOLORS Added To PICKMON TERMDEF File

The PICKMON terminal definition has a new boolean value added, XPICKCOLOR. This is a non-standard terminfo value (hence the name beginning with "X"). It allows the color definitions 'setab' and 'setaf' from PICK-style color sequences rather than ANSI style color sequences.

Note: Customers using PICKMON terminal definitions with a terminal client such as Accuterm that use the color sequences may need to manually upgrade their TERMINFO definition for PICKMON when they upgrade from earlier versions. New installations are not affected. This involves adding the following line to the PICKMON item in the TERMDEFS file

```
XPICKCOLORS,
```

and then running the program "COMPILE.TERM". Restarting the MV shell will activate the new definition.

3.2.7 SQL Changes

3.2.7.1 Improved SQL Support for Streams

This version of Caché has improved the handling of streams as fields in SQL tables, namely:

- It is now possible to open the stream returned via a simple function that returns of OID value of the stream. This is true for stream fields which are NULL as well, though attempting to read them will result in an immediate EOF.
- The INSERT and UPDATE clauses now accept such an OID as the new value for a stream field in a table.
- The SUBSTRING function now accepts a stream as its first argument.
- The INSERT and UPDATE clauses now accept such an OID as the new value for a stream field in a table. The SUBSTRING function now accepts a stream as its first argument.
- The ODBC and JDBC gateways now provide this support as well.

3.2.7.2 Support For SQL Column-Level Privileges Added

Caché SQL now supports column level privileges for SQL INSERT, UPDATE, SELECT, and REFERENCE privileges. The syntax for granting privileges at the column level for INSERT is:

```
GRANT INSERT ( FieldName1 [,FieldNameN ...] ) ...
```

for example,

```
GRANT INSERT ( Name, SSN ) ON Sample.Person TO John
```

will give John permission to insert into the Sample.Person table but only specify values for the Name and SSN fields. All other columns would be set to their default values (if they have one) or to NULL (if they have no default and are nullable).

The syntax for granting UPDATE and SELECT privileges is similar.

Caché SQL now also supports the ability to REVOKE privileges on columns. The syntax for REVOKE is the same as previously except that a user must specify the optional column list after the privilege. For example:

```
REVOKE INSERT (SSN) ON Sample.Person FROM John
```

Other items of note:

- Although Caché allows granting and revoking column level privileges for the REFERENCE privilege also, the REFERENCE privilege is not currently fully supported in Caché SQL. Full Support for REFERENCES is planned for the future.
- Column-level security does not invalidate tabel level privileges; these are still fully supported in a backward compatible manner. If an application uses only table level privileges, and has no need for column level privileges, everything will work as before.
- If a user grants a privilege at the table level, the SQL standard says the grantee will have privileges on all columns of the table as it is currently defined, and all privileges on any columns added to the table in the future. This is supported. However, Caché SQL does have the following limitation: You cannot grant a privilege at the table level, and then revoke the privilege from a set of columns of the table. Instead, the user should grant permissions only on the columns needed, leaving the remainder of the table inaccessible.
- The %CHECKPRIV statement has been enhanced to support checking for privileges at the column level.
- Users will now need SELECT privilege on any field names specified in an INSERT or UPDATE statement that are not part of the INSERT or UPDATE field list. In addition, If arrow syntax is involved, the user also needs SELECT privilege on the ID of the referenced table in addition to the referenced field.

3.2.7.3 Text Search In Long Streams

Previous versions of Caché provided the ability to search text represented as streams. However, searching streams longer than 32K required that long-string support be enabled. In this version of Caché that restriction has largely been lifted except for:

- Use of the %CONTAINS predicate requires that the stemmed, noiseword-filtered text be converted to a string. To avoid the possibility of that text being to long, applications should use the %CONTAINSTERM predicate instead.
- If the %SIMILARITY operator is used on a non-indexed field, the document will be broken up into chunks of 32K or less characters. In this case, terms that span boundaries may not be properly referenced. TO avoid this issue, %SIMILARITY should only be used on indexed fields.

3.2.7.4 Computed Property Self-Reference

Beginning with this version, computed properties can use {*} to refer to the current property in the compute code. This allows for more easily transportable compute code. For example, the following would be displayed in Studio

Property FOO As %String [Calculated, SqlComputeCode = { s {*}="bar"}, SqlComputed];

3.2.7.5 Duplicate Fields In Update or Insert Statements No Longer Allowed

Starting with this release, an SQL INSERT or UPDATE statement that contains duplicate fields will now be reported as being in error when the statement is prepared or compiled. For example, the following statement will result in an SQL error code:

```
insert into sample.person (name, Name) values ('Dave','tom')
 insert into sample.person set name = 'Dave', Name = 'Tom'
 update sample.person (name, Name) values ('Dave','tom')
update sample.person set name = 'Dave', Name = 'Tom'
```

The new error code value is -377; its message is “Field name '%1' appears more than once in assignment list of insert or update statement”.

3.2.7.6 SQL COALESCE Function Requires Compatible Types

In SQL,when preparing or compiling a query that contains the COALESCE function, Caché will now return an error (SQLCODE = -378) if the datatypes of arguments are not compatible with each other. Compatibility is judged according to the ODBC type of the data. If the type are compatible, but different, the COALESCE function will return a value of the type with the highest precedence. All the numeric types are compatible and have the following precedence:

- DOUBLE

- NUMERIC
- BIGINT
- INTEGER
- SMALLINT
- TINYINT

For example, in a COALESCE function call with arguments of type TINYINT, INTEGER, and NUMERIC, the type of the column will be NUMERIC because NUMERIC has the highest precedence.

All of the other ODBC types are currently not compatible with other types, so they must be explicitly CAST in the function argument in order to mix types.

Note: Normalizing the type to the one with the highest common precedence may change the way numbers are displayed. Changing the apparent type from INTEGER to NUMERIC, for example, will display the results with decimal points.

3.2.7.7 SQL UNION Enforces Type Compatibility

In previous versions an SQL query such as

```
SELECT 0 AS var1 FROM Table1
UNION ALL
SELECT 0.1 AS var1 FROM table2
```

report INTEGER as the column type of *var1* to ODBC and JDBC. The 0.1 value would be sent to the client as an integer and the data truncated as a consequence.

In this version the SQL processor will examine all parts of the union and return the type of the highest precedence for each column. The order of precedence for Caché types is:

- VARCHAR
- DOUBLE
- NUMERIC
- BIGINT
- INTEGER
- SMALLINT
- TINYINT

Thus, the query

```
SELECT MyTinyIntField AS var1 FROM Table1
UNION ALL
SELECT MyIntegerField AS var1 FROM Table2
UNION ALL
SELECT MyNumericField AS var1 FROM Table3
```

will return type NUMERIC for the column since it has a higher precedence than TINYINT and INTEGER.

Note: Normalizing the type to the one with the highest common precedence may change the way values are displayed. Changing the apparent type from INTEGER to NUMERIC, for example, will display the results with decimal points.

At this time Caché will only use the type precedence in UNIONs as listed above. If other types are involved, no automatic type detection will be done. The type of the union fields with the highest precedence will be returned. Other types will be ignored. If a specific type is desired for the column, an explicit CAST statement must be used, for example,

```
SELECT CAST(MyStringField AS DATE) AS var1 FROM Table1
UNION ALL
SELECT MyDateField AS var1 FROM Table2
```

3.2.7.8 SQL Query Collation Results Now Depend On All UNIONs

Beginning wit the version of Caché, if a FROM view or subquery consists of a UNION, then the collation of a resulting field will depend on the collations of the corresponding field/expression in all the UNION legs, using EXACT (no collation) if they do not match. In previous versions, the collation was determined by the first UNION found.

3.2.7.9 Stream Coercion Not Allowed In SQL UPDATE

Starting with this version, an attempt to Prepare or Compile an UPDATE statement that sets a non-stream field equal to the contents of a stream field will now receive an error. There is no implicit datatype conversion supported for this case. For example:

```
UPDATE SQLUser.MyTable
     SET MyStringField = MyStreamField ...
```

will fail with SQLCODE = –303: No implicit conversion of Stream value to non-Stream field 'MyStringField' in UPDATE assignment is supported.

The proper way to perform this assignment is to convert the fields from Stream to String using the SUBSTRING function. Suppose the length of MyStringField is 2000 characters. The statement:

```
UPDATE SQLUser.MyTable
     SET MyStringField = SUBSTRING(MyStreamField, 1, 2000) ...
```

will set the MyStringField to the first 2000 characters of the MyStreamField.

3.2.7.10 Support For LAST_DAY Function Added

Caché SQL now supports the **LAST_DAY()** scalar expression function. The syntax is

```
LAST_DAY(<expr>)
```

where <expr> is a %Library.Date or %Library.TimeStamp value. The return value of the function is the number of the final day of the month that contains the date or timestamp supplied.

3.2.7.11 Support Field Collation Names Without Leading %

When specifying a columns collation function in SQL DDL, Caché now supports the name of the collation without the leading "%" . For example Caché now supports:

```
CREATE TABLE MyTable (NAME VARCHAR(50) COLLATE EXACT)
```

as being identical to

```
CREATE TABLE MyTable (NAME VARCHAR(50) COLLATE %EXACT)
```

3.2.7.12 TCP Keep Alive Interval

This version introduces a new SQL Configuration setting that controls the TCP keep-alive timeout for individual processes (rather than relying on the machine-wide setting). This makes it possible to detect lost connections due to network failures, VPN trouble, and so on in a more reliable fashion. If there is a network failure, Caché will throw a <DISCONNECT> error within the timeout period specified rather than the machine default (which is something like 60 minutes).

The interval value is accessed via the Management Portal at **[Home] > [Configuration] > [SQL Settings]** as the field, "TCP Keep Alive interval" . It can be set programmatically via the method, SetTCPKeepAlive in the $SYSTEM.SQL class..

Note: This feature is available only on Windows and Linux platforms.

3.2.7.13 Static Cursors May Return More Rows

The limit on the number of rows that can be return by static cursors has been expanded. In prior releases, it was 32K rows. Beginning with this version, it is 4GB.

3.2.7.14 Unamed SAVEPOINT Support Removed

The SAVEPOINT capability was introduced in Caché version 5.1. This change removes support for unnamed savepoints to enhance compatibility with the savepoint implementations of other vendors. Unamed savepoints are not widely used among Caché applications. Applications that do use the feature will need to be modified to specify savepoint names.

3.2.7.15 ORDER BY Not Allowed In View

In previous versions, if an ORDER BY clause was used with a view in an SQL expression, Caché would ignore the ORDER BY clause. Beginning in this version, use of the ORDER BY in that situation will generate SQL error -143: ORDER BY not valid in a view's query.

3.2.7.16 Compilation Checks Field Names In Maps

In prior versions, the class compiler did not complain about references to nonexistent properties in the CacheSQLStorage map. Now, a non-existent field that is part of an expression in a map subscript will be reported as an error during class compilation.

The compiler now also checks that all field references are of the form *schema.fieldname* and that the schema is correct.

3.2.7.17 Privilege Checking Extended To Functions Called From Queries

Beginning in this version, Caché SQL now checks to see if the user executing a query has the privilege to execute any procedures called as user-defined SQL functions in that query. The user must have **EXECUTE** privilege on the functions procedure in order to execute the SQL statement that calls it. For example, to execute the query:

```
SELECT A, B, MyFunction() from SQLUser.MyTable
```

the user will need **SELECT** privilege on the SQLUser.MyTable table and **EXECUTE** privilege on the **SQLUser.MyFunction** procedure.

Note: As with all other SQL privilege checking, this only applies to SQL prepared and executed via ODBC, JDBC, or any flavor of Dynamic SQL. It does not pertain to Objectscript embedded SQL, “&sql(...)” .

3.2.7.18 Long Fields In xDBC Queries

If a class defines a property of type %Library.String, and the MAXLEN for this property is greater than 16374, when this field is selected in a query over ODBC or JDBC, only the first 16374 characters of the field will be returned. To support data in a single field longer than 16374 characters, use a stream datatype in place of the string.

CAUTION: Current ODBC and JDBC drivers have differing limits on the maximum length of returned columns. For those used with Caché, customers should assume that the maximum length returned by a query is 4096 characters.

3.2.7.19 Change To Conversion Of SQLBINARY, SQLVARBINARY And SQLLONGVARBINARY

The behavior for conversions of SQLBINARY, SQLVARBINARY, and SQLLONGVARBINARY has changed. It now returns two character hex values for each byte of data returned as SQL_C_CHAR and SQL_C_WCHAR.

The algorithm used in prior releases could result in erroneous conversions and/or incorrect string lengths. This caused problems for WinSQL when displaying SQLLONGVARBINARY columns. This change alters the behavior when using SQLFetch and SQLGetData for columns using these datatypes. The behavior now matches what is seen with SQLServer data.

3.2.8 CSP Changes

3.2.8.1 Properties Of %request Marked Read-Only

Modifying some properties of the %request object that are provided by the CSP gateway would cause the next request in the same process to also show this modified value. This could result in subtle errors in the application. To ensure this problem does not occur, this version of Caché now marks the following %CSP.Request properties as read only: UserAgent, ContentType, CharSet, Method, Protocol, Secure, GatewayApplication, GatewayConnectionName, GatewayBuild, GatewaySessionCookie, GatewayInstanceName, CSPGatewayRequest,

3.2.8.2 Added %request Properties

This version of Caché adds the following properties to the %CSP.Request object: GatewaySessionCookie and GatewayInstanceName.

3.2.8.3 Hyperevents Now Changed To Use xmlHttpRequest

In prior releases, CSP supported hyperevents in Java via CSP Java Event Broker and the JavaScript framework using iframes to invoke server methods. This was because not all browsers provided the xmlHttpRequest object for this purpose. This is no longer true. All supported browsers provide this functionality; it has been the default mechanism for CSP for several releases.

Beginning with this release, support for the older mechanism has been removed and all server requests will use the xmlHttpRequest for hyper-event processing. Specifically, this means:

- The methods used to include code for iframe and Java applet support are deprecated.
- The Java Event Broker and iframes have been removed from the product.
- The method calls **HyperEventFrame** and **HyperEventBody** of class %CSP.Page

  now always return the empty string.
- The method call **HyperEventBody** of class %CSP.Response

now always return the empty string.

- The attributes InsertBrokerIframe and InsertBrokerApplet of the CSP:CLASS tag are deprecated. When present, the javascript (.js) files for XMLHttpRequest hyperevent will be included instead.
- To avoid requiring changes to existing applications, the %CSP.Page) methods, HyperEventBody, and HyperEventFrame, will be retained and always return the empty string.
- The "HyperEvent Implementation" choice has been removed from the Management Portal.
- The property, HyperEvent, in the Security.Applications class has been deprecated. Applications using the security API to manipulate CSP pages should be changed since this code no longer has any effect. This property will be removed in a future release.

Note: InterSystems expects this change to have no impact on existing applications. The documented APIs for using hyperevent calls in CSP shield applications from the internals of the mechanism used to execute such calls so this change will be transparent to applications.

3.2.8.4 CSP Gateway Changes

The CSP Gateway consists of two modules: A Management Module, CSPmsSys.dll, and a Run-time, CSPms.dll. The run time is responsible for processing requests for CSP files and the Management module provides the Gateway's Management interface.

In older versions of the Gateway, these two modules were configured separately in the web server configuration. In this version, the run-time assumes responsibility for loading and routing management requests to the management module; it is no longer necessary to configure the Management module in the web server. This greatly simplifies the overall web server configuration for the Gateway.

Tip: To find the build of the CSP Gateway that you are running, in the System Management Portal, go to **[Home] > [Configuration] > [CSP Gateway Management]** and choose "About CSP" . The build number is the last four digits of the Gateway Build field, after the dot, such as 1020.

3.2.8.5 Serve Static Files From Caché Instead Of WebServer

In earlier versions, only files of type .csp, .cls and .zen were processed in Caché by the CSP engine. All other (such as static) files were served by the web server. With this version, the CSP engine is capable of serving any type of file that is placed in the CSP applications path (including static files). Among the advantages of this approach are these:

- The application is simpler to administer and deploy since all the files are on the Cache server. There is no need to copy static files to the web server machine; all files now come from the same place.
- This provides the ability to remap file locations so common files may appear to be in every path when there is a single copy in a central location in the /csp/broker application.

- Caché security rules can now be applied consistently both dynamic and static files.
- Tthe web server configuration for CSP applications to be further simplified because it is no longer necessary to create aliases in the web server configuration to represent the locations where the static files for an application reside. This resolves issues of contention when a single (i.e. common) web server serves two different versions of Caché, each requiring different versions of certain static files (for example, hyperevent broker components).

To support these changes, new configuration options specified with the CSP application settings have been added:

- Time to cache static files: Ths defaults to 1 hour if not specified. This is both the time the browser will cache this file and also the time the CSP gateway will cache this file if this caching is turned on.
- Modify the "Server Files" question to change the choices to:
  - No: Never server files from this application path
  - Always: Always server files from this application path, ignoring CSP security settings for this path for static files. This is the default for new applications; it is backward compatible with previous behavior serving files from the web server.
  - Always and cached: Always server files from this application path and allow the CSP gateway to cache these files to avoid having to request them from Caché. This is the mode most deployed application should use.
  - Use CSP security: If the user has permissions to view a .csp or .cls page in this application, allow them to view tatic files, If they do not have permissions to view a .csp or .cls page then return an error code of 404 (page not found) page. This is not cached in the gateway as only the server can decide if the user has the correct security permissions.

Any files placed in /csp/broker will effectively appear in all directories as the server will first look to see if the file, /csp/user/file.txt, is present. If not, it will look in /csp/broker/file.txt and, if there is a file with this name there, then it will serve this file up. This will allow applications to remove hard links to /csp/broker files which makes having a single server that serves up applications from multiple versions of Caché possible.

3.2.8.6 Password Change From Empty Detected

In previous versions, an attempt to change passwords using the CSP password change page where the old password was "" did not succeed. The code acted as if no old password had been passed to it. Now, it does detect that the old password was passed, was just "" and calls the password change code.

3.2.9 XML Changes

3.2.9.1 Support for Binary SOAP Messages

If the parameter SOAPBINARY is set to 1 for a web service, then binary transmission of SOAP messages between Caché instances will be enabled. The web service will then support normal XML based SOAP or Caché proprietary SOAP format over HTTP. The SOAP binary transport will be implemented using a custom object serialization mechanism. Classes used by the web service must be a subclass of %XML.Adaptor.

The WSDL produced for a Cache web service with SOAPBINARY parameter specified as 1 has been enhanced to carry the necessary information for the Cache web client. These WSDL extensions are valid according to the XML Schema, WSDL and WS-I Basic Profile specifications and are expected to be ignored by all conforming web client toolkits.

Note: Web Client toolkits which do not support the WS-I Basic Profile may experience problems with a WSDL that is generated for a Cache web service with SOAPBINARY = 1. This should be quite rare and found only with older toolkits.

3.2.9.2 Support XML Exclusive Canonicalization Version 1.0

Caché now supports XML Exclusive Canonicalization Version 1.0 (except for the InclusiveNamespaces PrefixList feature of Exclusive Canonicalization). This means the **Canonicalize** method of %XML.Writer writes the XML document represented by the subtree at a %XML.Node class instance in canonicalized form as mandated by the specification. As part of this implementation the order of output of attributes by the **Tree**, **Document** and **DocumentNode** methods is changed to be in the canonical order. This change will mostly be noticed in the output of schemas and WSDLs.

Note: This reordering will have no effect on any standards conformant XML processor.

As part of this implementation the SuppressAutoPrefix property is added to %XML.Writer and %XML.Namespaces which allows optional suppression of the prefix that is created for the default XML namespace even if it is not needed for the current element.

The **WriteChars**, **Document**, **DocumentNode** and **Tree** methods of %XML.Writer are modified to escape the characters as specified by the XML Canonicalization standard. The escaping is:

- “&” becomes “&”
- “<” becomes “<”
- “>” becomes “>”
- $CHAR(13) becomes “#xD”

and the **WriteAttribute**, **Document**, **DocumentNode** and **Tree** methods of %XML.Writer are modified to escape the characters as:

- “&” becomes “&”
- “<” becomes “<”
- “>” becomes “>”
- $CHAR(34), the double-quote character, becomes “"”
- $CHAR(9) becomes “#x9”
- $CHAR(10) becomes “#xA”
- $CHAR(13) becomes “#xD”

The change of escaping of text and attributes could be noticed, but should make no difference in any XML processing. The most noticeable change is that the **WriteChars** method of %XML.Writer will now escape $CHAR(13) characters and thus $CHAR(13,10) will have the $CHAR(13) escaped as #xD; which means that the $CHAR(13) will be in the data when it is later imported. Previously, it was lost.

3.2.10 Caché Terminal Changes

3.2.10.1 Cache Terminal Local Network Encoding Changed

The Caché Terminal local connection default network encoding has been changed to UTF8 in 8-bit installs to be consistent with the default Caché server default value.

3.2.10.2 Changes To Terminal Connection Display When Invoked From Scripts

Beginning with version 5.1, a Caché terminal session would begin by displaying the Node and Instance name that was the endpoint of the connection. It was done so a customer telneting to a machine would get a confirmation indication that they connected to the correct instance. However, this caused some existing customer scripts to fail because the display of the node and instance could throw off the parsing of the output.

In this version, the terminal will not display the node and instance if a routine is passed in on the command line, for example,

```
csession cache -U%SYS ALL^%FREECNT
```

since the process entering Caché is already on the correct node, and specifies the correct instance on the command line. In all other circumstances, the node and instance names will be displayed.

3.2.11 Standardize UNIX Names For ODBC

InterSystems is standardizing on iODBC for building our narrow and wide ODBC dependent projects. Cgate is already using an "i" or "u" suffix to designate iODBC and unixODBC respectively. A "w" suffix will designate a wide or unicode version of the library.

The file, libcacheodbc, will continue to be the narrow version of the InterSystems ODBC driver using iODBC headers, but libcacheodbciw will be the Unicode version of the InterSystems ODBC driver supporting iODBC.

Currently the narrow version of cgate and libcacheodbc will remain the default working executables in shipped examples. It is up the user to configure to use the Unicode versions of cgateiw or libcacheodbciw if needed.

Note: Mac platforms require a special link operation on libcacheodbciw to support the cppbinding and that will be called libcacheodbcmiw.

3.2.12 Language Binding Changes

3.2.12.1 Light C++ Binding Reference Count Behavior Changed

Prior to this change, if openid() was called twice for the same class and id, and the results assigned to two different lc_d_ref's, those lc_d_ref's pointed to two separate projection objects in memory, each initialized with property values from the same database object. Setting new property values using one would not affect the other. If changes made to one were saved, and then changes made to the other were saved, the property values from the last-saved projection object would overwrite those from the previously-saved projection object.

Similarly, if a projection object returned by create_new() was assigned to an lc_d_ref and saved to the database, and openid() was then called for that class with the id of the newly-saved object, and the result assigned to a second lc_d_ref, the two lc_d_ref's would point to two separate projection objects in memory.

This version changes the semantics of the Light C++ Binding object references (lc_d_ref's) to match the semantics of regular C++ binding object references (d_ref's), for cases in which multiple lc_d_ref's refer to an object of the same class with the same id.

Note: Well-designed applications should not experience any problem or notice any change in behavior. However, if an application was coded to use multiple references to the same object, and counted on the ability to modify the object through one of the references without affecting the in-memory values seen through the other references, the application's behavior will change. All of the references to the same class/id now point to the same object in memory.

3.3 Operators

3.3.1 System Management Portal

There have been no incompatible changes to the operator interface. Operators are advised to review the administrator section.

3.3.2 Default Configuration File Named Handling Changed

The default configuration file name used when Caché is started has changed. Please refer to the administrator section for the details.

4

Caché 2008.1 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2008.1 that, because of their difference in this version, affect the administration, operation, or development activities of existing 2007.1 systems.

General upgrade issues are mentioned at the start of this document. Those customers upgrading their applications from releases earlier than 2007.1 are strongly urged to read the upgrade checklist for earlier versions first. This document addresses only the differences between 2007.1 and 2008.1.

4.1 Announcements

4.1.1 Server Support For CPUs Before Pentium P4

Beginning with version 2008.1, Caché on Intel-based platforms will only install and run on servers using the Pentium P4 or later chipset, that is, those that support the SSE2 cpu extensions. This also applies to other manufacturers equivalent chipsets, for example, those from Advanced Micro Devices (AMD) that are functionally equivalent to the Intel Pentium P4.

Note: This is *only* for server systems. Caché will still run as a client on earlier CPU versions.

4.2 Administrators

This section contains information of interest to those who are familiar with administering prior versions of Caché and wish to learn what is new or different in this area for version 2008.1. The items listed

here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

4.2.1 Version Interoperability

A new table showing the interoperability of recent Caché releases has been added to the Supported Platforms document.

4.2.2 Management Portal Changes

4.2.2.1 Long String Control

The location for enabling use of long strings has changed. In prior versions, it was located on the **[Home] > [Configuration] > [Advanced Settings]** page. In this version of Caché, it has been moved to **[Home] > [Configuration] > [Memory and Startup]**.

4.2.3 Operational Changes

4.2.3.1 Journal Rollover Changes

Simple journal rollovers, regardless of their issuers, are no longer logged in journal history global (^%SYS("JOURNAL","HISTORY")). They are still logged in cconsole.log, however.

Journaling always starts in the primary directory, the primary directory is current directory and the secondary, the alternate. That remains the case until journaling gets an error (say disk full) in current (primary) directory and fails over to the alternate (secondary) directory, at which point the secondary directory becomes current directory and the primary, the alternate.

4.2.3.2 Maximum Path For Caché Databases Reduced

In this version of Caché, the maximum path length used to identify Caché databases and other items in the installation directory has been reduced from 232 to 227 characters. Customers who were close to the old limit may need to adjust some pathnames to fit within the new limit. To allow for variability in Caché file names, the directory portion of the path should be no longer than 195 characters.

CAUTION: If you are installing this version of Caché on a cluster configuration, you will need to:

1. Find the pathname of the directory holding the PIJ (Pre-Image Journal) files on each node of the cluster. This is the directory specified as the PIJ directory in the Clusters category in the .cpf file or the Management Portal page at **[Home] > [Configuration] > [Advanced Settings]**.
2. Shutdown the entire cluster cleanly.
3. Delete the PIJ files on each node of the cluster. These are files whose names are of the form "*.PIJ*.*" located in the PIJ directory for that node.
4. Upgrade each of the nodes to this version;
5. Reboot and re-form the cluster.

4.2.4 Parameter File Name Always Defaults to cache.cpf

The default for the optional configuration argument used when starting Caché from the command line has changed. Previously, when a command line such as

```
ccontrol start an_instance_name a_config_file
```

was issued to start Caché, a_config_file.cpf would become the default configuration file used to start later instances.

Beginning with this version, the configuration file will be used only for the current startup. Subsequent start commands which do not explicitly specify a configuration file will cause the Caché instance to be started using the cache.cpf configuration file.

4.2.5 Platform-specific Items

This section holds items of interest to users of specific platforms.

4.2.5.1 Windows Vista

Older Communication Protocols Not Supported

Due to underlying platform considerations, Caché on Windows Vista does not support DCP on raw Ethernet. Similarly, Caché DDP and LAT are not supported on this platform.

4.2.5.2 Mac

This version of Caché removes support for Apple Mac OS X for PowerPC.

4.2.5.3 Sun Solaris On SPARC-64

This version of Caché does not support the C++ and Light C++ bindings on this platform. It also does not support multithreaded CALLIN.

4.2.5.4 SUSE Linux Enterprise For Itanium

This version of Caché does not support the C++ and Light C++ bindings on this platform.

4.2.6 Limitations

The following known limitations exist in this release of Caché:

4.2.6.1 DCP Limitations

The following known limitations exist when using DCP:

- You cannot use the **^%RCHANGE** utility over DCP — a `NETWORK DATA UPDATE FAILED` error occurs.
- The System Management Portal does not correctly show mapping of databases connected through DCP.

4.2.6.2 DDP Limitations

The following known limitations exist when using DDP:

- You cannot use the **^%GD** utility over DDP — a `<DIRECTORY>` error occurs.
- Caché Studio does not work over DDP.
- XML import and export do not work over DDP.
- Global lengths cannot be longer than 757 bytes over DDP.

4.2.6.3 ODBC Limitations

ODBC clients from versions prior to Caché 5.1 are not fully compatible with this version. InterSystems recommends that the client systems be upgraded.

4.3 Developers

This section contains information of interest to those who have designed, developed and maintained applications running on prior versions of Caché.

The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

4.3.1 ObjectScript Changes

4.3.1.1 $REPLACE Function Added

This version of Caché adds a new string replacement function, $REPLACE. **$REPLACE(S1, S2, S3)** finds all occurrences of the string S2 in the string S1 and replaces them with the string S3. All three arguments are required. Unlike **$TRANSLATE**, S2 and S3 are treated as complete strings, not lists of characters.

4.3.1.2 $ZDATETIMEH Validation Improved

In this version, ODBC and SQL date-time validation is improved. $ZDATETIMEH will now report an error if any field (day, month, hour, second) is not in the proper format. For example, 002 will no longer be accepted as a valid day within a month because only two digits are allowed.

4.3.2 SQL Changes

4.3.2.1 Incompatibility: “arrow syntax” in ON clauses

Prior to Version 2007.1, Caché SQL allowed use of arrow syntax in ON clauses. This was possible because Caché only supported a linear chain of left outer joins (one or more tables left joined to a single table). As a result, combinations of arrow syntax, uses of =* syntax, and ANSI joins could readily be mixed within this very limited support.

Arrow syntax has always been defined as equivalent to =* syntax, and uses the same underlying mechanism. As a result, it can no longer be supported ON clauses.

All other support for arrow syntax remains (WHERE clause, SELECT clause, GROUP BY, ORDER BY, HAVING).

4.3.2.2 Restriction on LEFT JOIN

There remains a restriction with ON clauses with LEFT JOINs. If all the conditions affecting some table use comparisons that may pass null values, and that table is itself a target of an outer join, Caché may report:

```
error 94: unsupported use of outer join
```

An example of such a query is

```
SELECT *
FROM Table1
          LEFT JOIN Table2 ON Table1.k = Table2.k
          LEFT JOIN Table3 ON coalesce(Table1.k,Table2.k) = Table3.k
```

4.3.2.3 Collation Order Now Applies to LIKE

The SQL LIKE operator now performs similarly to other comparison operators such as "=" and %STARTSWITH. That is, the expression:

```
f LIKE <arg>
```

is now interpreted as:

```
Collation(f) LIKE Collation(<arg>)
```

where Coll() is the default collation of f. (Unlike the other operators, in this case the default collation of <arg> does not affect the result.)

This applies to fields with default collation: %SQLSTRING, %SQLUPPER, %UPPER, and %SPACE. It does not apply to %MVR, %PLUS, and %MINUS. It also does not apply to %ALPHAUP and %STRING (because these may remove characters from a string).

Note: The exclusion of %ALPHAUP and %STRING will be changed in a future version.

For collations with truncation, such as %SQLUPPER(50), the collation (and truncation) are applied to the field value and to the pattern. In practice, this means that the pattern used should be shorter than the truncation length, otherwise some cases could yield non-intuitive results.

4.3.2.4 COMMIT Behavior Changed To Match SQL Standard

The behavior of COMMIT when there are multiple active SAVEPOINTs was modified to conform to the SQL standard. For a pure SQL application, the COMMIT action will commit all nested transactions up to and including the most recent BEGIN TRANSACTION. More generally, for a mixed COS/SQL application where the COS code could have created additional nested transaction levels, a SQL COMMIT will commit all nested transactions up to and including the highest $TLEVEL that was not created by an SQL SAVEPOINT statement.

CAUTION: As has always been the case, applications that call out of SQL and increase *$TLEVEL* should generally restore the *$TLEVEL* before returning to SQL. Such applications should not end a transaction that was started by SQL, otherwise SQL-conformant behavior cannot be guaranteed.

4.3.3 TSQL Changes

4.3.3.1 %TSQL.Manager.load() Replaced

With this release, the **load()** of %TSQL.Manager has been deprecated. The new, preferred method for loading TSQL source files is **$SYSTEM.SQL.TSQL()**. This function accepts arguments similar to that of the new %TSQL.Manager.**Import()** method.

4.3.3.2 TSQL Procedure Definitions Map to Default Schema

When executing TSQL batches using $SYSTEM.SQL.**TSQL()** or **##class(%TSQL.Manager).Import()**, the classes created to contain procedure definitions will now be in the package mapped to the default schema. For example, if the default schema is 'dbo' then the package in which the procedure classes will be created will be the package defined with the SQLNAME = 'dbo'.

4.3.3.3 Support For Delimited Ids

This version of Caché adds a checkbox to the SQL Gateway connection setup that prevents using delimited ids in the SQL statements sent over the Gateway. This option was added because Sybase does not support delimited ids; connections that do not select this option will fail if they connect to Sybase databases.

4.3.4 Command-line Debugging Changes

The **ZBREAK** command has been extended with new arguments that permit finer control over instruction stepping. For example, it is now possible to bypass stepping into language support routines and method calls, particularly the %Destruct method.

Details on the improved operation are contained in the **ZBREAK** command reference material, and the section on debugging in the guide to Caché ObjectScript.

4.3.5 Studio Changes

4.3.5.1 Source Control Document Compatibility

The %Studio.SourceControl.Interface class implementation has changed in this version of Caché. However, Studio will now detect which version of the class is being used and act accordingly. This will allow more controlled conversion of existing applications using source control in Caché.

4.3.5.2 Version Compatibility

This version of Studio will refuse connections to servers running versions of Caché before 5.2. This is a consequence of the security features added in that release. When a connection is refused, developers will see a message, "Version mismatch. Server should be version 5.2 or higher."

4.3.6 Caché Terminal Changes

4.3.6.1 Terminal Identification

In this version, the TERMINAL has been modified to display the machine and instance of Caché it is connecting to. These items are displayed BEFORE any prompt for a user ID or password occurs. Those terminal scripts that assume the first output from TERMINAL is the prompt for a username should be changed to explicitly look for the prompt. Otherwise, they may intermittently fail depending on the timing of the output.

4.3.6.2 Support For Telnet Terminal Types

Caché Terminal now supports the "Terminal Type" telnet option. At process startup, Caché will perform the "Terminal Type" telnet option negotiation. If the telnet client agrees and sends the terminal type, Caché will define the Windows *TERM* environment variable. The value of this variable can be retrieved from within Caché through the function, **$System.Util.GetEnviron("TERM")**.

4.3.6.3 ZWELCOME

This version of Caché provides a mechanism to invoke specific code to be executed when a terminal connection is made, but before the user login takes place.

The terminal initiation code checks for the existence of a routine called, **ZWELCOME** in the %SYS namespace. If such a routine is found, it is invoked immediately prior to the terminal login sequence, if any. The name of the routine implies its intended use, as a custom identification and welcome message to users.

CAUTION: The **ZWELCOME** routine executes in the %SYS namespace with an empty *$USERNAME* and *$ROLES* set to **%ALL**. Care should be taken to ensure that the failure modes of **ZWELCOME** are benign.

4.3.7 Class Changes

4.3.7.1 %Library.String

The **LogicalToDisplay** method has been changed in this version. In converting the internal form to an external representation, it removes all NULL ($CHAR(0)) characters from the string. This contradicts the usual expectation that **LogicalToDisplay** and **DisplayToLogical** are inverses of one another.

4.3.7.2 %XML.Document and %XML.Node

This version of Caché introduces the %XML.Document class that represents an XML document as a Document Object Model (DOM). The DOM may be created either

- from an XML document by accessing the Document property of %XML.Reader after calling an Openxxx method of %XML.Reader.
- as a new instance by calling the **CreateDocument** method to instantiate an empty document.

The document can be written to a destination with the various **Writer** and **Tree** methods of %XML.DOM.

The %XML.Node class is used to access the components of the DOM. %XML.Node navigates through the nodes of the DOM rather than representing a fixed node in a DOM tree. The **MoveToxxx** methods are used to move through the DOM. The properties and methods of %XML.Node are then used to retrieve and modify the node contents.

Note: A set of macros in the file, %xmlDOM.inc may also be used to navigate the DOM based on the DocumentId property of %XML.Document.

4.3.7.3 New $SYSTEM.SQL Functions

Two new functions have been added to this class:

- **$SYSTEM.SQL.UserExists(UserId)** returns 1 if a user named <UserId> exists, and 0 otherwise.
- **$SYSTEM.SQL.RoleExists(RoleId)** returns 1 if a role named <RoleId> exists, and 0 otherwise.

4.3.7.4 MANAGEDEXTENT Class Parameter Added

A new data management mechanism has been implemented in this release. Because of this, those classes that use the default storage mechanism for managing their data, %Library.CacheStorage now behave differently with regard to their persistent instances.

The globals used by persistent classes that utilize default storage are now registered with the new Extent Manager; the interface to the Extent Manager is through the %ExtentMgr.Util class. This registration process occurs during class compilation. Any errors or name collisions are reported as errors, causing the compile to fail. Collisions must be resolved by the user. This can done by either changing the name of the index or adding explicit storage locations for the data.

The extent metadata is only deleted when the class is deleted. To delete the extent using Studio, right-click on the name of the class in the **Workspace** window of Caché Studio, and select **Delete Class '<classname>'** from the menu with the "e" flag is set as the default for the namespace.

The available flags and qualifiers are shown set by the commands

```
Do $SYSTEM.OBJ.ShowFlags()
Do $SYSTEM.OBJ.ShowQualifiers()
Do $SYSTEM.OBJ.SetFlags()
Do $SYSTEM.OBJ.SetQualifiers()
```

Recompiling a class will refresh the extent metadata. A failed compile leaves the metadata in the state it was in at the time of the failure.

The user can always explicitly delete the extent metadata using **##class(%ExtentMgr.Util).DeleteExtent(<classname>)**.

If the user does not want to register the global references used by a class then set the value of the MANAGEDEXTENT class parameter to 0 (zero).

Note: It is possible that an application has been designed with multiple classes intentionally sharing a global reference. In this case, the implementer will need to add MANAGEDEXTENT=0 such classes, if they use default storage. Otherwise, recompilation of an application in the set will generate the error like

```
ERROR #5564: Storage reference: '^This.App.Global used in 'User.ClassA.cls'
is already registered for use by 'User.ClassB.cls'
```

4.3.7.5 %Zen.Report

Report in %ZEN has been changed to correct a number issues discovered in earlier versions and to extend some capabilities. A summary follows.

OnCreateResultSet Callback

For reports that use the OnCreateResultSet callback, this callback has been enhanced so that it is now passed an array of user-defined parameters. For example, the method declaration now is:

```
ClassMethod RS1(ByRef pSC As %Status,  ByRef pParameters) As %ResultSet
```

so that the result set request

```
<group name="city"
       breakOnField="Home_City"
       OnCreateResultSet="RS1"
       parameter field="Home_City"/>
```

so that pParameters(1) will now contain the current value of field Home_City, etc. Prior to this change, there was no mechanism to pass parameters into user-created result sets.

Nested queries

This version corrects an issue with nested queries. A Zen Report can either define one outer query and have the grouping levels break on columns within this query, or it can introduce additional queries at the group level. In this latter case, the nested query is typically fed some parameter from the outer query, for example:

```
<report sql="SELECT City FROM Table1 ORDER BY City">
<group name="City"
       breakOnField="City"
       sql="SELECT Employee FROM Table2 WHERE City=?">
<parameter field="City"/>
```

In prior versions, many cases of nested query did not work correctly. Now nested queries are executed only *once* for each new break value from the outer grouping.

In addition, the internal management of nested queries is now more consistent in how references to fields within queries are resolved. The basic rules are that:

1. Every group (including the outer report tag) defines a "query context" at the "level" of the group. The level specifies how deeply nested the group is from the top of the report definition. If a group does not define a new query, then it uses its the query of its parent group as if it was its own.
2. References to fields within a <group>,<parameter>, or <attribute>node are resolved by looking at the query of the parent node.
3. References to fields within <element> and <aggregate>nodes are resolved by looking at the query at the same level as the node.

For example, in

```
<report sql="SELECT Name FROM Table1">
<element name="A" field="Name"/>
```

Name comes from Table1. In

```
<eport sql="SELECT Name FROM Table1">
<ttribute name="A" field="Name"/>
```

Name cannot be resolved and an error message is generated. In the sequence

```
<report sql="SELECT Name FROM Table1">
<group name="Name" sql="SELECT Name FROM Table2 WHERE...">
<element name="A" field="Name"/>
```

Name will be resolved to that in Table2. And finally, given

```
<report sql="SELECT Name FROM Table1">
<group name="Name" sql="SELECT Name FROM Table2 WHERE...">
<element name="A" field="Name"/>
```

Name will be resolved to Table1.

Non-Existent Fields

References to non-existent fields within a query now result in “” for the value of the field and not the value, “not found” .

Sibling Groups

It is now possible for a ReportDefinition to define more than one group node at the same level. These are referred to as "sibling" groups and are descibed in more detail in the Zen Reports documentation. There are some special rules in effect for sibling groups:

- Column break logic only applies to the first sibling-- breakOnField and breakOnExpression are ignored for any group that is not the first sibling.
- Aggregates are only computed if they appear within the first sibling.

Sibling groups are typically used when each sibling defines its own query, usually referring to a breaking field from the outer query in their common WHERE clause. They are also used where the siblings do not define their own queries. In this case, the first sibling tests for break conditions and outputs its records, then the subsequent siblings are processed with the same break field.

Node Level

As a convenience, the Report Engine defines a variable, *%node(level)* to be equal to the current number at the given grouping level. You can use this within an expression within a ReportDefinition. For example,

```
<attribute expression="$G(%node(2))" name="num"/>
```

Important: The effect of these changes is that some previously acceptable queries will now be reported as being in error.

4.4 Operators

4.4.1 System Management Portal

There have been a number of changes and improvements to the system management portal since 2007.1 was released. These are detailed in the administrator section.

4.4.2 Default Configuration File Named Handling Changed

The default configuration file name used when Caché is started has changed. Please refer to the administrator section for the details.

5

Caché 2007.1 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 2007.1 that, because of their difference in this version, affect the administration, operation, or development activities of existing 5.2 systems.

General upgrade issues are mentioned at the start of this document. Those customers upgrading their applications from releases earlier than 5.2 are strongly urged to read the upgrade checklist for earlier versions first. This document addresses only the differences between 2007.1 and 5.2.

5.1 Administrators

This section contains information of interest to those who are familiar with administering prior versions of Caché and wish to learn what is new or different in this area for version 2007.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

5.1.1 New Distribution Format

Starting with this release, InterSystems is changing the distribution medium for Caché from CDROM to DVD.

CAUTION: You must have a CD reader capable of reading DVDs in order to install this version of Caché.

5.1.2 System Management Portal Changes

In Caché 2007.1, InterSystems has modified and improved the version of the System Management Portal available with Caché 5.2. A summary of the more important changes follows.

5.1.2.1 EnableLongStrings Configuration Parameter

Caché allocates a fixed amount of space to hold the results of string operations, the string stack. If a string expression exceeds the amount of space allocated, a <STRINGSTACK> error results. For existing applications, this should not be a problem. However, since this release increases the maximum size of Caché strings, applications that attempt to use this long strings may experience problems if this is not enabled.

When turned on, the configuration parameter, EnableLongStrings, located at **[Home] > [Configuration] > [Advanced Settings]** under the Miscellaneous category, increases the size of the string stack by approximately 50 times to accommodate operations involving larger strings.

This switch is a system-wide setting. A Caché job checks this switch when it starts to see whether it should use long strings or not, and allocates a large (support long strings) or normal (do not support long strings) string stack accordingly. Although a program may change the switch setting using **$zu(69, 69)**, it will not affect the current job. Only subsequent jobs will use the changed setting. Once a job allocates a string stack, it cannot change the size later in the same job.

Note: Enabling this configuration parameter may affect Caché performance. Memory is often a precious resource and allocating it to the stack comes at the expense of other uses. Retuning of the system may be needed to maintain performance.

Note: If this parameter is not enabled, then the default maximum length for read operations will remain at 32,767 characters.

5.1.3 Changes in Default Port Choices

This version of Caché chooses the default ports for installation differently from perevious releases. The new algorithm is:

Superserver

If port 1972 is unused, choose it. Otherwise, choose the first available port number equal to or greater than 56773.

WebServer

Choose the first available port number equal to or greater than 57772.

With this change, Caché now conforms to RFC793.

5.1.4 Increased Maximum String Length

In this release, the maximum length of a Caché string has been increased from 32,767 to 3,641,144 characters. This change is transparent to existing applications. However, those applications that are modified to handle larger strings should consider enabling the EnableLongStrings configuration parameter.

Note: This limit applies only to databases with 8KB blocks. For older databases with a 2KB block size, the maximum remains unchanged: 32767 characters.

The maximum string size limit applies only to the size of the resulting string. Other size-related limits remain as before. For example, the length of a line of code given to the compiler is still 4096 characters. The maximum size of routines, determined by a configuration parameter, is 32K or 64K. So the length and number of string literals remains little changed.

5.1.5 Improvements in Routine Handling

This version of Caché has significant improvements in the way COS routines are managed. The changes are detailed in Routine Performance Enhancements.

5.1.6 Caché Terminal Uses Authentication Setting

In this version of Caché if authentication is enabled (for example, by installing Caché with Normal security), Terminal sessions will start with an authentication prompt such as requesting the OS userid and password.

5.1.7 System API Changes

5.1.7.1 %SYS.LDAP Added

This class, %SYS.LDAP, is provided to assist management of systems using the Lightweight Directory Access Protocol (LDAP) by providing an ObjectScript interface to an LDAP database. Callers can use the methods provided to authenticate themselves to the database. Once authenticated, they can add and delete items from the LDAP database, and search for information managed by LDAP.

5.1.7.2 Changes to Journaling API

An API, %SYS.JournalFile, has been implemented for purging all journal files except those required for transaction rollbacks or crash recovery:

```
##class(%SYS.Journal.File).PurgeAll()
```

A crash recovery refers to the journal restore performed as part of Cache startup or cluster failover recovery.

WARNING! Post-backup journal files are not necessarily preserved. Therefore, if you want to be able to restore databases from backups and subsequent journal files, you should configure the journal purging parameter based on backups and use the regular purging API (PURGE^JRNUTIL).

There is also an API for purging journal files based on criteria different from what is in the Cache configuration:

```
##class(%SYS.Journal.File).Purge(NDaysOld As %Integer,
                                 NBackupsOld As %Integer) returns %Status
```

This method purges old journal files based on criteria given taking care not to purge files required for transaction rollbacks or crash recovery. The parameters are:

- NDaysOld – journal files must be at least this number of days old to be purged
- NBackupsOld – journal files must be at least this numberof backups old to be purged

If both parameters are specified, only one criterion has to be met (inclusive OR) to qualify a journal file for purging subject to the restriction about rollback and crash recovery.

5.1.7.3 Generic Memory Heap Configuration API Added

A new API, %SYSTEM.Config.SharedMemoryHeap, has been added to assist in gathering information about how Caché uses the heap. It also provides the ability to get available generic memory heap and recommended gmheap parameter for configuration. For example, the **DisplayUsage** method displays all memory used by each of the system components and the amount of available heap memory:

```
Write $SYSTEM.Config.SharedMemoryHeap.DisplayUsage()
```

The **RecommendedSize** method returns the recommended amount of heap memory that should be configured in this instance for smooth operation:

```
Write $SYSTEM.Config.SharedMemoryHeap.RecommendedSize()
```

Note: This does not include the memory reserved for shadowing when enabled which is (2 * #CPUs + 1)MB, for example, A 4–CPU system will require an additional 9MB.

5.1.7.4 IntegrityList Method Removed

The method, **IntegrityList**, has been removed from the class, %SYS.GlobalQuery because it duplicates functionality already available. Customer may use one of the other existing methods to accomplish the same objective:

Classname	Method	Description
%SYS.GlobalEdit	CheckIntegrity	Check the integrity of a single global (instance method)
%SYS.GlobalEdit	CheckGlobalIntegrity	Check the integrity of a single global (class method)
SYS.Database	CheckIntegrity	Check the integrity of a single database (instance method)
SYS.Database	IntegrityCheck	Check the integrity of a single database (instance method)
SYS.Database	Integrity	Performs integrity check over all databases (same as the removed method)

5.1.8 Platform-specific Items

This section holds items of interest to users of specific platforms.

5.1.8.1 Windows

Installer Changes

- With this release, the default installation directory for Windows has been changed to C:\InterSystems\Cache.
- For compatibility reasons, the installer no longer allows Caché to be installed into the Program Files directory.
- In conjunction with the change to the default installation directory, the installer now appends the entry, %SystemRoot%\InterSystems\Common Files\intersystems\cache to the end of the system-wide environment variable, *Path*. This directory contains Caché-specific DLLs.

Changes for Vista

- *Operating system upgrade requires a new Caché installation* — Windows Vista introduces significant upward compatibility issues for programs. Because of this, upgrading to Vista on systems with existing Caché installations will almost certainly cause Caché to malfunction.

  InterSystems requires that customers wishing to use Vista install new versions of Caché. Existing data and applications can be moved via save and restore of the relevant databases. The application should be recompiled.

- *Security adjustments* — Depending on the options chosen, some additional manual adjustments may need to be made after installation of Caché. See the "Installing Caché on Microsoft Windows" chapter of the *Caché Installation Guide* for details.
- *Protocol limitations* — Caché does not support DDP, LAT, or DCP over raw Ethernet under Window Vista. Please consult the Supported Platforms document for details.
- *CacheODBC log default location changed* — the default location for the log is now under %PUBLIC%\Logs\CacheODBC.log. This directory is accessible by all users and allows just one location for the log to be created. The old location was under %WINDIR%. On Vista, this is virtualized to C:\users\someuser\AppData\Local\VirtualStore\Windows\CacheODBC.log. The result is that there would be a different location for each user, making the log difficult to find for support issues.

  The log file location can be changed by setting the new location into the environment variable *CACHEODBCTRACEFILE*.
- *Startup encryption option differs slightly on Windows Vista* — if an instance uses database encryption and supports Startup with interactive key activation, Caché prompts for encryption key information. This prompt causes Vista to display an Interactive services dialog detection dialog box. The window for this dialog is initially minimized, so that it may not be visible if you configure Vista to hide the taskbar. See the Database Encryption chapter of the Caché Security Administration Guide for details.

5.1.8.2 Use Large Page Size for Shared Memory – Windows Server SP1

Caché for Windows has been enhanced to make use of Windows large pages for the shared memory section. Large pages are a hardware feature which Windows makes accessible to software applications when allocating shared memory sections starting with Server 2003, SP1. The actual size of the page depends on the processor used. The recognized processors and their large pages sizes are:

- EMT 64–bit: 2MB
- AMD 64–bit: 2MB
- Itanium: 16MB

Caché will attempt to use large pages whenever the feature is available from Windows and the process starting Caché can acquire the SeLockMemoryPrivilege resource. Performance is enhanced on systems which allocate large buffer pools because they are locked into physical memory and they make more efficient use of the hardware translation buffers. Since large pages are locked into physical memory care must be taken not to request too large a buffer pool; this may not leave enough physical memory for the number of users the machine is intended to support. This condition, not leaving enough memory, could result in worse performance than if the buffer pool had been used normal sized pages.

On Windows this calculation (how much physical memory to leave for users) must take into account the Page Table Entries which Windows creates as part of its memory management. Page Table Entries

(PTEs) consume 4 bytes each on a 32-bit system and 8 bytes on 64-bit system. Windows allocates one page table entry for every 4KB of memory. This means that on a 64-bit system dividing the size of the shared memory section by 500 gives approximately the # of bytes Windows allocates for PTEs to map the shared memory section. This memory is private to each process; each process needs to allocate this chunk of memory.

The management of these PTEs can itself become a performance issue since these PTEs are pageable. Care must be taken to ensure that physical memory is available for these PTEs. Problems here may not show up at first as the PTEs can be paged out until the shared memory they reference is accessed so a system may appear to be functioning properly but become dramatically worse as time goes on and demand increases.

On most platforms when Caché starts up and fails to allocate the requested shared memory for global and routine buffers, it retries with a series of requests decreasing the amount of space requested with each iteration. This continues until it either succeeds or reaches the limit on the number of retries. If the latter condition occurs, Caché logs a message and startup aborts.

On Windows the approach occurs in two phases. Caché will first try using large pages, assuming large pages are supported. The startup process attempts to acquire the necessary privilege and run through the request loop. If it succeeds, then it proceeds to use large pages for memory managing memory. If it fails attempting to use large pages, the loop is restarted from the original memory request without using large pages. If the 2nd iteration through the loop fails, Caché fails to start.

5.1.8.3 Remote Database Pathnames

In prior versions, upgrading would change the drive letter for remote databases to uppercase. This is no longer done. Those sites with Windows ECP servers should examine the pathnames for remote databases in **[Home] > [Configuration] > [Remote Databases]**. Those whose drive letter is in uppercase should be changed to lowercase.

5.1.8.4 UNIX / Linux / Mac OS X

- Daemon Privileges

  In versions of Caché prior to 5.1, daemons always ran as root, and Caché used that privilege to set the UNIX parameter RLIMIT_FSIZE to unlimited. However, since 5.1, the Caché daemons may run under a different userid. Since only root may set its own limits, the limits cannot be set for a non-root owner.

  In addition, when running as a non-root user, Caché failed to set the current (soft) limit to the hard limit. In this case, even if the RLIMIT_FSIZE hard limit was unlimited, if Caché was started from a login user whose soft limit was set to some lower value, the Caché daemons ran with this lower value, and could encounter serious I/O errors attempting to write to CACHE.DAT or CACHE.WIJ.

  Cache now checks that the RLIMIT_FSIZE hard limit is unlimited, and halts the cache startup (or installation) with an error message if this is not the case. Second, it sets the soft limit to the hard limit in the daemons.

- CDL Support Dropped for Mac OS X

  In this version of Caché, CDL support has been removed from the Mac platform. Please use XML for transport of applications across platforms.

5.1.8.5 Feature: xDBC

Client applications attempting to connect to server systems running this version of Caché must be running on Caché version 5.0.13 or later. Attempts to connect from earlier versions will fail.

Please contact the InterSystems Worldwide Response Center (WRC) if this is an issue for your site.

5.1.8.6 Dreamweaver No Longer Supported

The release of Caché no longer supports any version of Dreamweaver as a method for generating CSP pages.

5.1.8.7 Python Binding Version Requirements

For Caché version 2007.1 on Windows, customers wishing to use the Python language binding should use ActiveState Python 2.4 with Microsoft Visual Studio 7. InterSystems has tested the product specifically with ActiveState Python 2.4.3.12.

5.2 Developers

This section contains information of interest to those who have designed, developed and maintained applications running on prior versions of Caché.

The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

5.2.1 System Management Portal

There have been a number of changes and improvements to the system management portal since 5.2 was released. These are detailed in the administrator section.

5.2.2 SQL Changes

5.2.2.1 Caché SQL NOT! and NOT& Operators Removed

In this release of Caché, the FDBMS-era operators NOT! and NOT& have been fully removed from the SQL syntax. Please contact the InterSystems Worldwide Response Center (WRC) if this is an issue for your applications.

5.2.2.2 SYSDATE Now Reserved

As an assistance to customer migrating their applications to Caché. SQL now supports the function, **SYSDATE**. This is equivalent to the SQL standard function, **CURRENT_TIMESTAMP**, but the latter is preferred.

Note: **SYSDATE** is now an SQL reserved word. Applications using SYSDATE as an identifier must switch to using the delimited identifier syntax, "SYSDATE" , or change the name of the identifier, to resolve the conflict.

5.2.2.3 CREATE TABLE Changes Default Schema Assignment

In Caché SQL, there is no default package defined with a schema name of USER. An attempt to create a table with a schema name of USER when no package definition for USER exists will create a package mapping with a package and a schema name of USER.

This overrides the default mapping of the USER package to the SQLUser (schema) mapping.

In addition, USER is an SQL reserved word, but since there was no ambiguity in previous releases, the SQL parser allowed USER as a schema name.

In Caché 2007.1, USER is no longer allowed as the schema name in DDL statements unless it is a delimited identifier. This prevents ad hoc DDL development of tables, or other SQL objects, from creating a package mapping that overrides the Caché default mapping of User->SQLUser. Undelimited uses of USER will result in the new SQL error

```
SQLCODE = -312: Invalid schema name 'USER'
```

5.2.2.4 Oracle Timestamps and JDBC Gateway

When linking to any column defined in an Oracle database as an Oracle Date, the linking procedure incorrectly represents this as a Caché Date. This interpretation makes it impossible manipulate data items declared as Date (Timestamps, actually) via JDBC.

To obtain the correct behavior, once the table has been linked, it is necessary for the user to open the class definition in Studio, and change the datatype from %Date to %TimeStamp.

This is a documented issue with the Oracle JDBC drivers.

5.2.2.5 TuneTable and KeepUpToDate Flag

Caché provides the ability to gather statistics about the contents of a table to assist in optimizing queries. The method that can be invoked by applications at runtime is: **$SYSTEM.SQL.TuneTable**.

The fifth parameter of **$SYSTEM.SQL.TuneTable** is the KeepUpToDate flag. If this flag is set to 1, Caché does not mark the classes and tables as out of date when it modifies the values for ExtentSize and the Selectivity.

In Version 2007.1, KeepUpToDate has been extended to influence cached queries. If KeepUpToDate is set to 1, Caché will not purge cached queries as part of the TuneTable.

If the flag is 0, the default, then the class is marked as not up to date and all cached queries based on the table are purged. Running TuneTable on a live system with KeepUpToDate = 0 can cause application errors if the application is using any queries that are removed from the cache.

5.2.2.6 Changes for Sybase / SQL Server Compatibility

The following changes have been made to Caché for compatibility with Sybase and SQL Server:

- the command, TRUNCATE TABLE, has been added
- six new functions are now available for use in expressions: CHARINDEX, DATALENGTH, REPLACE, STUFF, and SYSDATE
- two additions have been made to the list of reserved words: SYSDATE, and STATISTICS

5.2.2.7 New DDL Datatype Mappings

As of this version, the following new datatype mappings have been added to the system set of datatype maps:

External Type	Maps In Caché To
BINARY VARYING	%Library.Binary(MAXLEN=1)
BINARY	%Library.Binary(MAXLEN=1)
CHAR VARYING	%Library.String(MAXLEN=1)
CHARACTER VARYING	%Library.String(MAXLEN=1)
NATIONAL CHAR VARYING	%Library.String(MAXLEN=1)
NATIONAL CHARACTER VARYING	%Library.String(MAXLEN=1)
NATIONAL VARCHAR	%Library.String(MAXLEN=1)
VARBINARY	%Library.Binary(MAXLEN=1)
VARCHAR	%Library.String(MAXLEN=1)

5.2.2.8 Statement Changes

The following is a short list of changes to statement parsing. Please refer to the SQL Reference manual for further information.

- The INTO keyword is now optional on an INSERT statement
- The UPDATE and DELETE statements now allow a new FROM clause to support multi-table selection criteria
- The FROM keyword is now optional in a DELETE statement

5.2.2.9 Operator / Predicate Changes

The following are new in version 2007.1:

- The LIKE predicate supports letter case collation
- %CONTAINSTERM has been added as a new comparison operator

5.2.2.10 New Comment Syntax

Caché SQL now supports multiline comments with the same /* ... */ syntax used by Caché Objectscript.

5.2.3 Changes to %Text Class Stemming

The %Text.English class uses the Porter stemmer, which is a widely used published algorithm for stemming English words. The published algorithm includes an amended method for stemming "-ed" word endings that has been used in all versions of the %Text.English class.

In this version of Caché, the algorithm has been changed to restore the behavior of the originally published algorithm, because the amended algorithm does an inferior job on most words ending in "-ed" . This change affects what terms go into an index based on the %Text.English class, so any index that is not rebuilt across versions may experience slight differences with respect to text predicates on words ending in "-ed" .

Important: It is recommended that you rebuild %Text.English text indices that were created on prior versions of Caché. Otherwise, a word that could end in "-ed" might fail to match a different form of the word already in the index.

5.2.4 Changes to Callin

In this release of Caché, CALLIN no longer supports msql* functions. These must be changed to Caché SQL.

5.2.5 Corrections

5.2.5.1 String Truncation

An error has been fixed that under some circumstances caused Caché to ignore maximum string length settings. This allowed strings whose length exceeded the maximum to be accepted instead of being detected as being too long. Subsequent use, for example, when the value was passed to a stored procedure or persisted, would correctly truncate the value. Values are now processed consistently for both the initial and subsequent uses.

5.3 Operators

5.3.1 System Management Portal

There have been a number of changes and improvements to the system management portal since 5.2 was released. These are detailed in the administrator section.

6

Caché 5.2 Upgrade Checklist

Purpose

The purpose of this section is to highlight those features of Caché 5.2 that, because of their difference in this version, affect the administration, operation, or development activities of existing 5.1 systems.

Those customers upgrading their applications from releases earlier than 5.1 are strongly urged to read the upgrade checklist for Caché 5.1 first. This document addresses only the differences between 5.1 and 5.2.

6.1 Upgrading from Field Test Versions

Customers running on any prior released version of Caché may upgrade to this version of Caché during installation.

CAUTION: InterSystems does not support an upgrade from any of the versions used for field test of this release. This includes any versions of Caché distributed to customers at DevCon or other events.

6.2 Administrators

This section contains information of interest to those who are familiar with administering prior versions of Caché and wish to learn what is new or different in this area for version 5.2. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

6.2.1 System Management Portal Changes

In Caché 5.2, InterSystems has modified and improved the initial version of the System Management Portal available with Caché 5.1. A summary of the more important changes follows.

6.2.1.1 Control Of Encryption

The **[Home] > [Database Encryption]** page is now the central location for managing encryption of databases and journal files. If the administrator chooses to encrypt a database, there is an additional prompt asking whether to encrypt the journal files as well.

Encrypted databases may now automatically be mounted at startup provided that access to the keyfile data is available.

6.2.1.2 Secure Communication Using SSL & TLS

Caché 5.2 allows secure communications using the Secure Sockets Layer (SSL) and Transport-Level Security (TLS). The available configurations for these protocols can be accessed via the SSL./TLS Configurations choice on the **[Home] > [Security Management]** page. Administrators also have the options of creating their own configurations.

6.2.1.3 Database Directory Management

The Local Databases and Remote Databases choices on **[Home] > [Configuration]** page now allow the administrator to identify the location of a directory containing a Caché database. The database directory is now a link that allows the user to bring up a Directory Browse window to be used in locating the new location.

Once the administrator selects a directory and confirms the choice, The portal will check to see if the new directory actually contains a CACHE.DAT file. If CACHE.DAT does exists, then the database edit page will be refreshed with the new location along with property values loaded from this new directory. Once the administrator hits the Save button, the database information will be saved to the configuration file.

6.2.1.4 Editing Globals

The **[Home] > [Globals] > [Edit Global Data]** page now allows the administrator or operator to edit and delete global data values.

6.2.1.5 Language Project Settings

The Java, C++ and EJB Projection settings in the **[Home] > [Configuration] > [Advanced Settings]** page have all been removed. Application developers should use Studio to manipulate these settings.

6.2.1.6 Granting Roles

The **[Home] > [Security Management] > [Roles] > [Edit Role]** page now allows roles to be granted to users. It has been extended to allow granting of the current role to other roles, and granting other roles to the current role.

6.2.2 CSP Session Data Moved

The data about the current CSP session, *%cspSession*, in Caché 5.2 is mapped to CacheTemp. In previous releases, it was mapped to CACHESYS. The new mapping has two main effects:

First, CSP session data is no longer journaled. Depending on system usage, this may bring a noticeable reduction in the size of the journal.

Second, because it is located in CacheTemp, the CSP session data no longer survives a system interruption and the session cannot be resumed after a system restart. It also means that the OnSessionEnd event will not be fired in the event of a system interruption, and therefore temporary data may not be properly disposed after restart.

The previous behavior can be restored by setting the configuration parameter, JournalcspSessions, in **[Home] > [Configuration] > [Advanced Settings]** to 1.

Note: Any existing %cspSession globals will be removed during an upgrade to Caché 5.2.

6.2.3 Platform-Specific Items

This appendix holds items of interest to users of specific platforms.

6.2.3.1 Feature: TSQL

Support for TSQL is available only on the following platforms at the time of the release of Caché 5.2:

- Windows – 32–bit only
- LINUX
  - Red Hat– 32–bit only
  - SUSE – 32–bit only
- AIX – 64–bit only
- Solaris SPARC – 64–bit only

Please contact the InterSystems Worldwide Response Center (WRC) if this is an issue for your site.

6.2.3.2 Windows Clusters

InterSystems recommendations for setting up Windows cluster configurations have evolved over the past several versions. Please check your configuration against the current recommended settings.

6.2.3.3 HPUX

SSL/TLS Support On 32–Bit HPUX 11

The Secure Sockets Layer (SSL) and Transport-Level Security (TLS) features of Caché 5.2 are not available on this version of HPUX.

6.2.3.4 SUSE Linux

SSL/TLS Support On Itanium

There is a known issue with the openssl-devel package supplied by InterSystems. Users should remove the InterSystems-supplied package and install the version that comes with the operating system instead.

6.3 Developers

This section contains information of interest to those who have designed, developed and maintained applications running on prior versions of Caché, particularly those that have made the transition to Caché 5.2.

The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

6.3.1 System Management Portal

There have been a number of changes and improvements to the system management portal since 5.1 was released. These are detailed in the administrator section.

6.3.2 IEEE 8-byte Floating Point Support

As noted in the Release Notes, this version of Caché now supports operations on floating-point data in the IEEE-754 format. The Objectscript function that does this conversion is **$DOUBLE**; the Caché datatype class for it is %Library.Double.

In addition, the SQL SystemDataType mapping has been changed. In previous releases, the SQL datatypes DOUBLE and DOUBLE PRECISION were mapped to %Library.Float (and implemented as ObjectScript numbers). In this release, they are mapped to %Library.Double (and implemented using the **$DOUBLE** function).

The major difference between the two representations lies in the achievable numeric precision. Caché represents numbers to 18 decimal digits of precision. IEEE-754 uses a binary mantissa of 53 bits, resulting in a decimal precision of 15 decimal digits (actually about 15.3). There are two effects of this that developers must be aware of:

- Those applications obtaining floating-point data via an ODBC or JDBC interface will see the exact value of the number that was transmitted in their application. Previously, the conversion to an ObjectScript numeric value may have resulted in a slightly different value being used internally.
- The conversion of ObjectScript numeric values to IEEE-754 will result in a loss of precision. Whether this materially affects the application depends on the sensitivity of the calculations to this effect.

Note: The range of values allowed for $DOUBLE in Caché 5.2 is:

Maximum: 1.79769e+308

Minimum (normalized): 2.22508e-308

Minimum (denormalized): 4.9e-324

6.3.2.1 Mixed Usage – $DOUBLE & Caché Decimal

Those who intend to introduce $DOUBLE into existing calculations must be aware that several factors may influence the accuracy of the resulting calculation:

Conversion Between Representations

The maximum value of a Caché decimal number is 9.223372036854775807e145. Conversion from DOUBLE to Caché decimal values will result in errors if the numbers involved are out of the Caché range.

Conversions from Caché decimal values to DOUBLE values always succeed.

Large Numeric Literals

The ObjectScript parser has been optimized to recognize large numeric literals beyond the range of Caché decimal values only when supplied as the argument to the $DOUBLE function. Thus,

```
set x = 1.0e200
```

results in a <MAXNUMBER> error while the following statement succeeds:

```
set x = $DOUBLE(1.0e200)
```

Testing For Equality

Since DOUBLE values employs a binary floating-point format, the conversion of a decimal number with a fractional part to DOUBLE may produce an inexact representation of the original number.

Therefore, comparisons between two calculated DOUBLE values for equality may not produce the expected result.

A better approach is to test for the difference of two values being within some tolerance. For example, if A and B are two DOUBLE values, and EPSILON is the tolerance, the test should be written as

```
IF ($ZABS(A - B) < EPSILON)
```

6.3.2.2 DOUBLE With ODBC/JDBC

CAUTION: If a Caché version 5.2 server contains tables with columns mapped as DOUBLE, attempts to access those columns by clients running on earlier versions of Caché will fail. Such clients must also be running under Caché 5.2.

6.3.2.3 SQL LEFT JOIN Changes

Because of improvements to the way OUTER JOINs are handled in this release, two instances of questionable syntax and semantics that previously had been allowed are now reported as errors:

- The sequence, “=*” , is no longer allowed in ON clauses. In some cases, this had the effect of causing an INNER JOIN to be interpreted as a LEFT JOIN.
- Arrow syntax (->) is no longer accepted in ON clauses. Although the documentation always indicated that the only acceptable syntax was “field1 = field2” , this was sometimes accepted due to poor error checking during expression validation

6.4 Operators

6.4.1 System Management Portal

There have been a number of changes and improvements to the system management portal since 5.1 was released. These are detailed in the administrator section.

7

Caché 5.1 Upgrade Checklist

Purpose

The purpose of this chapter is to highlight those features of Caché 5.1 that, because of their difference in this version, affect the administration, operation, or development activities of existing systems.

Background

Caché version 5.1 is a significant improvement in functionality and security over its predecessors. In making this advance, InterSystems goal was to provide a compatible evolutionary path forward whenever possible. However, many of the features, such as the System Management Portal, replace functions in previous releases with new mechanisms. Furthermore, the addition of the new security features required a partial redesign and reorganization of the underlying system. These introduced incompatibilities with previous versions of Caché.

Other Resources

Other InterSystems documents describe the features of Caché 5.1 in more depth and breadth. For example,

- The Getting Started With Caché section of the documentation Home page provides the Release Notes for this release, information on Installing Caché, and a list of the target Supported Platforms.
- The Caché System Administration section contains information on Administering Caché including using the new System Management Portal, and also Administering Caché Advanced Security.
- A new section called Caché System References provides two new books. One describes the format of the Caché Parameter File. A second, the Caché Advanced Configuration Settings Reference, explains the parameters in the **[Home] > [Configuration] > [Advanced Settings]** page. It also shows where those settings were found in the Caché version 5.0 user interface.

7.1 Administrators

This section contains information of interest to those who are familiar with administering prior versions of Caché and wish to learn what is new or different in this area for version 5.1. The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

7.1.1 New License Keys Required

Note: Caché version 5.1 introduces new capabilities and a new key format. The license servers from prior releases and the license server for Caché 5.1 do not recognize each others key formats. Existing users MUST obtain new licenses from InterSystems in order to run Caché version 5.1. Please contact your local sales representative to obtain the new keys corresponding to your existing license or to discuss new licensing options.

If a site wishes to run a 5.1 installation on a system where 5.0.x systems will run concurrently, the systems must obtain the correct license keys from their respective servers. This is done by individually setting the ports to the license servers on each system. Caché 5.0.x systems default the license server port to 4001. The license server port for the version 5.1 system(s) should use a different port number for accessing their license server.

Multiple Caché instances that share a key must all be upgraded to 5.1 together.

7.1.2 Recompilation After Upgrade

As noted in the Release Notes, and elsewhere in this document, the changes made to Caché for this version are extensive and pervasive.

Important: All user application classes must be recompiled after upgrading to version 5.1. And, all user routines that contain embedded SQL statements must also be recompiled as well.

CAUTION: Failure to recompile after upgrade may result in unexplained failures during application execution, and possible data loss.

7.1.3 System Management Portal

Prior to this version, how Caché was administered depended heavily on the platform where Caché ran. With version 5.1, InterSystems introduces a new administrative interface that is common across all platforms. Caché 5.1 now uses a browser-based interface, the System Management Portal, for system management.

An advantage of this approach is that (which few exceptions) it is no longer a requirement that any Caché component be installed on the system you use to manage an installation. Remote management of systems over the web, subject to access control established for the site, is now possible and easy. Cross-release compatibility issues are eliminated because both the data and its formatting information come directly from the system being managed.

This new interface subsumes the functions previously distributed among Explorer, SQL Manager, Configuration Manager, and Control Panel functions of the Windows Caché Cube. Because it combines these functions, operators and some developers will also use the portal to accomplish their tasks as well.

Important: The version 5.1 management portal cannot be used to manage earlier versions of Caché. The opposite is also true; the management functions of earlier versions cannot be used to manage Caché configurations running version 5.1.

More information on the System Management Portal can be found in the System Administrator documentation.

7.1.3.1 Portal and Application Name Conflicts

In Caché 5.1, the instance name chosen at installation time, is used to construct the name of the CSP application that runs the System Management Portal. For example, assume a Caché system had a CSP application called, "/appserver" . If the installation name chosen for this system was "APPSERVER" , the upgrade procedures would construct a CSP application to run the System Management Portal called, "/appserver/csp/sys" . After the upgrade this would effectively block access to the previously available CSP application.

Note: When upgrading from an earlier version, care must be taken to ensure that there is not already a CSP application with the same name as the installation (ignoring differences in case).

7.1.4 Security Advisor

To assist system managers in securing a Caché system, version 5.1 includes a Security Advisor. This utility shows current security-related system configuration information, recommends changes or areas for review, and provides links into other system management Web pages to make recommended changes.

Caché 5.1 contains the initial version of the Security Advisor. Its function and range will expand in future versions. It is accessed through the System Management Portal at **[Home] > [Security Management] > [Security Advisor]**.

InterSystems strongly recommends a review and resolution of the issues raised by the Security Advisor before allowing a secured system to attain production status.

7.1.5 Defaults for Security Settings at Installation Time

The use of Caché security begins with the installation (or upgrade) of version 5.1. During Caché installation, the person doing the installation is prompted to select one of three initial security settings:

- Minimal
- Normal
- Locked Down

The selection determines the initial configuration settings for Caché services as follows:

Security Setting	Minimal	Normal	Locked Down
User authentication required	No	Yes	Yes
Password pattern default	3.32ANP	3.32ANP	8.32ANP
The _SYSTEM user is enabled	Yes	Yes	No
Roles assigned to **UnknownUser**	%All	<None>	<None>
Create installation username and prompt for password	No	Yes	Yes

And the following table shows what services are enabled by default:

Service Name	Minimal	Normal	Locked Down
%Service_Bindings	Yes	Yes	No
%Service_CSP	Yes	Yes	Yes
%Service_CacheDirect	Yes	No	No
%Service_CallIn	Yes	No	No
%Service_ComPort	No	No	No
%Service_Console	Yes	Yes	Yes
%Service_DCP	No	No	No
%Service_DDP	No	No	No
%Service_ECP	No	No	No
%Service_LAT	No	No	No

Service Name	Minimal	Normal	Locked Down
%Service_MSMActivate	No	No	No
%Service_Monitor	No	No	No
%Service_Shadow	No	No	No
%Service_Telnet	No	No	No
%Service_Weblink	No	No	No

7.1.6 Emergency Access

As a contingency, Caché provides a special emergency access mode that can be used under certain dire circumstances, such as severe damage to security configuration information or "unavailability" of any users with the **%Admin_Manage:U** or **%Admin_Security:U** privileges. (Although Caché attempts to prevent this situation by ensuring that there is always at least one user with the **%All** role, that user may not be available or may have forgotten his or her password.)

When Caché is running in emergency access mode, only a single user (the "emergency user") is permitted. Caché is started in emergency access mode through a command-line switch, which passes a user name and password for the emergency user. This user name does not have to be previously defined within Caché. (In fact, even if the user name is defined in Caché, the emergency user is conceptually a different user.) The emergency user name and password are only valid for a single invocation of emergency mode.

The user starting Caché in emergency access mode must have operating-system level system management privileges. (On Windows systems, the user must be a member of the Administrators group. On UNIX systems, the user must be root. On OpenVMS systems, the user must have a system UIC.) Caché authenticates this user by checking his or her operating system level characteristics. When Caché is started in emergency access mode:

- The emergency user is the only permitted user. Any attempt by another user to log in will fail.
- The emergency user automatically has the **%All** role.
- The Console and CSP services are enabled. All other services are disabled. This does not affect the enabled or disabled status of services in the configuration; only the current "in memory" information about services is affected.
- Caché password authentication is used and unauthenticated access is forbidden for all services.
- If possible, auditing is enabled for all events. Caché startup proceeds even if this is not possible.

7.1.7 Configuration File Changes

One consequence of the new capabilities added in version 5.1 is that major differences have been made to the form and content of the configuration file that provides many of the initialization values when Caché starts up. This section does not detail every change in the configuration file, only the more apparent ones.

Note: As of this version, the parameter file MUST be named cache.cpf.

7.1.7.1 New Parameter Reference

If a site controls operation by editing the .cpf, each of these controls must be examined to make sure they are still applicable. Administrators are strongly urged to review the Caché Parameter File Reference book for the current organization, and the valid parameters and their allowed settings.

7.1.7.2 Startup Check for Configuration Changes

Caché configuration information is stored outside of Caché and (by design) can be modified when Caché is not running. Therefore, a special protective option have been added to this version. Rather than protecting the contents of the configuration file, Caché controls the ability to start the system or modify the configuration of a running system. This protection is enabled by turning Configuration Security on. (Of course, the configuration file can and should be protected outside of Caché by strictly limiting at the operating system level the ability of users to modify that file.)

During startup, if Caché detects that the configuration (.cpf) file has changed since the last time the Caché instance was started, the user running startup will be asked to enter a username and password. This data will be used to verify that the user is authorized to start Caché with altered configuration parameters. If the user is successfully authenticated, and the user has **`%Admin_Manage:Use`**, Caché will be started with the new configuration parameters.

Otherwise, Caché will start with the values of the last known configuration. When this happens, the configuration file supplied will be copied to cache.cpf_rejected (overwriting any file by that name), and the configuration parameters actually used to start Caché will be written to the file specified as the configuration file.

7.1.8 Low-level Management Interfaces

In addition to the new Management Portal, system administrators can exercise low-level control over the security of Caché systems through character-oriented utilities. The available routines are described in The CHUI-Based Management Routines.

7.1.9 Web Server Changes

The evolution of security capability in Caché 5.1 has affected the way hypertext information is served to browsers.

7.1.9.1 The Caché Private Web Server is Now Apache

Each installation of Caché 5.1 also installs an instance of Apache as its private web server. Sites may change the configuration to use a different one, but Caché will always install its private server regardless.

7.1.9.2 Default Port Changes

By default, Caché now chooses as the superserver port number the first unused port at or after 1972. Applications which depended on the value, 1972, may fail to contact the superserver. In addition, Caché now uses a separate port number for its private web server. The value chosen is the first unused port at or after 8972. Despite both ending in "72", the two port number are not correlated; the superserver port might be 1973 and the web server port number could be 8977.

During a custom installation, user may explicitly set both the superserver port and the private WebServer port numbers.

7.1.9.3 CSPGateway Changes

Caché version 5.1 has changed the CSPGateway implementation on two of the platforms.

Support Removed for OpenVMS

The CSPGateway is no longer supported on OpenVMS. The material referencing it has been removed from the OpenVMS installation script and the code is no longer part of the product for OpenVMS.

Note: Any existing CSP Gateway files will be removed during a upgrade from a previous version.

WebServer Parameter for OpenVMS

The OpenVMS .cpf files now contain the WebServer parameter again in the format

```
WebServer=[ON|OFF],<server>:<port>
```

If no such parameter is specified in the .cpf file, the defaults chosen are "OFF" , "" , and the first available port on or after 8972, respectively.

7.1.9.4 Default Username and Password

When the CSP gateway connects to Caché the first message it sends over is a login message that can contain a username and hashed password defined in the CSP gateway management pages. If the server

$username="" which means that CSP gateway did not connect with Kerberos, it will use the default username and password to attempt to login this service.

If this fails, the CSP gateway halts after recording an entry in the audit log (if auditing is enabled). If it succeeds, then $username will not be null and then the CSP server is allowed to call other functions such as ones to display CSP pages. While $username="" the CSP server will only call the login method.

7.1.10 Caché Permissions on UNIX/Linux are Those of the Installer

In prior versions of Caché, it was necessary to install Caché under the username, root. This was not a good practice because Caché does not require root privilege for normal operation. In version 5.1, this requirement has been eliminated.

When Caché starts, it now sets its userid to that of the user that installed it and its groupid to "cacheusr". One consequence of this is that devices which are inaccessible to that installer may also be inaccessible to Caché. If you wish this installation to have access to devices available only to *root*, Caché must be installed by *root*.

For example, on many UNIX systems *root* owns the Ethernet devices. A version of Caché installed by a non-root user would not (by default) be able to communicate using the Ethernet.

CAUTION: The "cacheusr" groupid must have write permission for the files needed to operate Caché, for example, the files in the Journal directory. Failure to meet this requirement will result in erratic operation possibly leading to system failure.

This warning extends to any file and/or directory used by Caché created by an administrator outside of Caché. For example, if the journal files are assigned to a separate disk to improve throughput and system reliability, it is not enough that they be created by a user with, for example, "root" access. They must be writable by the "cacheusr" group.

7.1.11 New Password Hash Function

Caché has always stored its password data for users as the result of applying a hash function to the characters of the password. When you attempt to login, a hash of the characters you enter is calculated and it is compared to the hashed value stored internally. If the two calculated values match, the passwords are assumed to match.

This version of Caché uses a computationally stronger function to compute the password hash; one that produces different hash values than before and is harder to "crack".

Because different password strings can hash to the same value, there is no way to compute the actual user's password starting from the hash value. This means there is no way to compute the new password hash value starting with the old hash value. Therefore, all userids on the system must be given new password hash values when upgrading to version 5.1.

User data exported by prior versions of Caché (for example those produced by **$SYSTEM.SQL.Export(...)**) contains the password hash values used for that version. Care must be taken when importing such data into version 5.1. All such users will need new passwords assigned. Users whose ids are imported will have their password hashes reset preventing them from logging in until it is reassigned under 5.1.

An exception to this is users who have null (no assigned) passwords. These users will be processed automatically.

Note: After an upgrade, Caché 5.1 will assist in the computation of the new password hash values. When a user attempts to login for the first time, the password hash will be calculated using the previous algorithm. This value will be compared against the stored value. If they match, the password hash will be recalculated using the new algorithm and this new value will be stored in the database. Thus the conversion of passwords will be made as existing users login for the first time.

7.1.12 Read-Only Databases

7.1.12.1 Representation

To improve consistency in handling read-only databases, the way they are identified to Caché has changed in this version. Caché now recognizes read-only databases that are marked as such in properties for that database, or through declared read-only via the **^MOUNT**.

7.1.12.2 Write Daemon Access Determines Database Mode

In Caché version 5.1, when a database is mounted the write daemon checks whether it has sufficient permission to update the database. If it does not, it will force the database to be mounted in read-only mode.

7.1.13 Cluster Changes

7.1.13.1 Improved Cluster Join Logic

Caché 5.1 has been enhanced so that a cluster member is no longer allowed to fully join a cluster while the trio of switches 10, 13, and 14 (which disables database access) is set on the cluster master. In prior releases, the new system would be allowed to cluster mount and read/write from databases. If the cluster was in the process of performing a backup, this could cause problems.

Now the new cluster member will detect the switches which have been set cluster-wide and set those switches locally while it starts up. This may mean that the Caché startup process on the member attempting to join the cluster will hang if a switch is set which blocks global access. A console log message will be generated if this occurs.

Note: This version can interoperate with older versions but the new functionality will not be present unless the master and the system joining the cluster have not been upgraded to version 5.1.

7.1.14 Journaling Changes

As a result of experience with prior versions of Caché, journaling in version 5.1 has been significantly improved as one part of the underlying support for highly available systems. The goal of the changes in 5.1 has been to make the behavior of journaling safer and more consistent; for example, journaling is now a property of databases rather than individual globals. The operator interface has been changed to incorporate this new approach, and Caché 5.1 provides for common management and auditing of changes to journal settings via the System Management Portal.

7.1.14.1 Journaling is Now a Database Attribute

Caché 5.1 sets the specification of the journal state on a databases basis. This greatly improves reliability of the system because it addresses inconsistencies (after crash recovery) that could arise in earlier versions due to change in globals that may or may not be journaled, and that may or may not be involved in transactions explicitly or implicitly via %Save or SQL UPDATE statements. The changes in more detail are:

- The journaling state is a property of databases, not individual globals. All globals within a database are journaled or not, depending on this setting. There are only two states - YES and No.
- The default setting of the journal state for new databases is YES. When a database from an earlier version is first mounted, the value is set to YES, regardless of the previous setting for "default for new globals" and regardless of the settings of individual globals within that database.
- In a transaction, Caché writes changes into the journal, regardless of the settings of the databases in which the affected globals reside. Rollback will work as before.
- Nothing mapped to CACHETEMP is ever journaled; its journaling behavior is unchanged.
- Journal restore respects the *current* settings of the database. Nothing is stored in the journal about the state of the database when the journal is written. The state of the database at time of restore determines what action is taken. This means that changes to databases with JOURNAL=YES will be durable, but changes to other databases may not be. Caché will ensure physical consistency, but not necessarily application consistency if transactions involved databases with JOURNAL=NO.
- Databases mounted on a cluster have their globals journaled or not, depending on the database setting.
- The setting for a database can be changed on a running system. If this is done, the administrator will be warned of the potential consequences and the change in state audited.

In recognition of this change, Caché 5.1 also:

- changed the default purging settings to somewhat mitigate the diskspace consequences of this change;
- removed the routine, **^%JOURNAL**, which in prior releases enabled or disabled journaling on a per-global basis;
- modified **^%GCREATE** and **^%SYS.GCREATE** so they no longer ask whether to journal globals.

CAUTION: One aspect of the new journal design is that restores are performed only to databases marked to be journaled at the time of a journal restore. The ^JRNRESTO program now checks the database journal state the first time it encounters each database and records the journal state. Journal records for databases not so marked are skipped during restore.

If no databases are marked as being journaled, the ^JRNRESTO program will ask if the operator wishes to terminate the restore. Administrators can change the database status to journaled and restart ^JRNRESTO if desired.

7.1.14.2 Journaling Z-Globals

In prior releases, the JournalZGlob parameter was used to indicate whether z/Z* globals should be excluded from journaling (even inside transactions). In version 5.1, to make journaling more robust, it has been removed. When upgrading an earlier Caché system with the flag set, the existing individual z/Z* globals in every defined database are given the journal attribute of that database. (For CACHETEMP, the journal attribute defaults to off).

If a site needs to exclude new z/Z* globals from journaling, the administrator will have to map z/Z* globals to a database with the journal attribute turned off.

Note: Since the globals in a namespace may be mapped into different databases, some may be journaled and some not. It is the journal setting for the database to which the global is mapped that determines how the global will be treated.

To replicate the behavior in prior versions, when the flag to exclude journaling of z/Z* globals is set, the z/Z* globals in every namespace must be mapped to the CACHETEMP database. The difference between CACHETEMP and a database with the journal attribute set to off is that nothing in CACHETEMP, not even transactional updates, gets journaled.

7.1.14.3 Changes to Journal Purge

Prior to this release, the default behavior was to purge the journal files after 7 days. In Caché version 5.1, the default has been changed.

You may have Caché purge journal files after either X days, or Y successful backups have occurred. Normal recommended settings are:

- 1 <= X <= 100
- 1 <= Y <= 10

If both X and Y are > 0, files will be purged after X days or Y successful backups, whichever comes first. If either X or Y is zero, purging is done on the basis of the remaining criteria. Setting X and Y to zero prevents purging entirely.

Journal files are now purged after 2 consecutive successful Caché backups.

Note: Those customers who do not use Caché backup facilities should consider scheduling the appropriate journal maintenance using, for example, the Caché Task Manager to manage the amount of journal information retained.

7.1.15 Shadowing Changes

This version of Caché has significantly improved facilities for system shadowing. It is better at shadowing to and from clusters. The latency reporting on both the sender and shadow systems have been improved and there is better control over suspending/resuming and starting/stopping shadowing.

7.1.15.1 Journal Applied Transactions to Shadow Removed

The setting to choose whether or not to journal applied transactions on shadow databases no longer exists. In earlier Caché releases the default behavior was to not journal updates to shadow databases; but you could enable journaling on the shadow by selecting the **Journal Applied Transactions** check box. This maintained a separate journal file showing the activity on the shadow database.

With the option removed in Caché 5.1, journaling of the shadow databases is determined by the global journal state of the databases themselves. After an upgrade, journaling on the shadow databases is enabled. To mitigate the increased demand on the storage capacity of the shadow, Caché purges the destination shadow copy of a source journal file once it is dejournaled and does not contain any transactions open on the shadow.

InterSystems recommends you journal all databases that are the destination of shadowing. However, if you do decide not to journal the destination shadow databases, you must also disable journaling on the CACHESYS database. Caché stores the journal address and journal file name of the journal record last processed by shadowing in the ^*SYS* global in the CACHESYS database. This serves as a checkpoint from which shadowing will resume if shadowing fails.

CAUTION: On the shadow destination, if you journal the CACHESYS database, but not the destination shadow databases, there is the possibility that if the shadow crashes and restarts, the checkpoint in CACHESYS could be recovered to a point in time which is *later* in the journal stream than the last record committed to the shadow databases.

7.1.15.2 Compatible Mode (Record Mode) Shadowing Removed

There is no longer an option to choose the method of journal transmission. All shadowing uses the *fast mode, apply changes* method.

Prior to Caché version 5.1, there were four methods of journal transmission for shadowing:

- Fast mode, apply changes
- Fast mode, don't apply changes
- Compatible mode, apply changes
- Compatible mode, scan changes

Compatible mode (previously called record mode) was most often used for compatibility among heterogeneous platforms, and sometimes to support different Caché releases. *Fast mode* (previously called block mode) now supports heterogeneous platforms since it automatically performs any necessary byte reordering for different endian systems.

If you wish to support multiple production servers running different Caché releases from a single shadow, then InterSystems recommends that you set up multiple Caché instances on the shadow server, one for each Caché version, and use *fast mode* rather than *compatible mode* on older versions. This provides the best performance and reliability.

Important: A Caché upgrade converts existing *compatible mode* shadows to *fast mode*. The converted *fast mode* shadows may or may not work with the sources, depending on the source configuration. Caché 5.1 automatically performs endian conversion for *fast mode* shadowing.

7.1.15.3 Changes in Shadowing Defaults

In Caché 5.1, the following databases are not shadowed by default:

- CACHEAUDIT
- CACHELIB
- DOCBOOK
- SAMPLES

You can click **Add** next to the **Database mapping for this shadow** list on the **[Home] > [Configuration] > [Shadow Server Settings] > [Edit Shadow Server]** page of the System Management Portal if you wish to shadow them.

7.1.16 CACHETEMP

Caché 5.1 handles CACHETEMP differently from its predecessors. The changes are a result of security requirements and customer requests.

7.1.16.1 Expansion and Size Characteristics Preserved

Caché 5.1 preserves the expansion and size settings of CACHETEMP across restarts. After restart, the size reported by Caché will the minimum of 240MB or the allocated size of the file, whichever is smaller. If the size of the file allocated by the operating system is larger than 240MB, Caché will only initialize the map blocks to describe the first 240MB and will expand the map later as needed. It will not, however, shrink the physical size of the file.

7.1.16.2 Collation

After restart, the collation of CACHETEMP will be reset to Caché Standard regardless of its prior setting. Those sites that wish a different collation should add code to the "SYSTEM" callback of the ^%ZSTART routine to set the collation desired.

7.1.16.3 Conditions for CACHETEMP Deletion

Under the following circumstances:

1. CACHETEMP is a 2KB database
2. CACHETEMP is mounted when STU (system startup) runs

Caché attempts to delete and recreate CACHETEMP. Condition #2 occurs, for example, if Caché is started in "nostu" mode, and then the operator later runs STU manually.

When CACHETEMP is recreated, the initial size is set to 1MB, the expansion factor to 0 (indicating growth by the larger of 10% or 10MB), and the maximum size to 0 (no limit).

7.1.17 ShutDownTimeout Parameter Now Enforced

Beginning with version 5.1, the ShutDownTimeout parameter will be enforced on all platforms. Shutdown will not spend more than the value of ShutdownTimeout (less about 10 seconds) in user-defined shutdown routines. Once the limit is reached, shutdown will proceed to completion (including final force cleanup) even if user-defined shutdown routines have not completed.

7.1.18 Collation for Locales Now on by Default

When a national collation is available in a locale (for example: Spanish1, Portuguese2, German2), it is now set as the default collation for that locale, instead of "Cache Standard". When a locale has more than one collation (such as German1 and German2), the one with the greatest suffix was selected.

Locales that don't have national collations (English, Hebrew, and so on), continue using "Cache Standard" as their default collation. The changes are summarized in the following tables:

Note: This affects only the creation of local arrays, because new globals have their collation taken from the database's default (unless explicitly created by %GCREATE).

Locales Whose Default Collation Changed

Locale	Collation
chsw	Chinese2
csy8	Czech2
csyw	Czech2
dan8	Danish1
danw	Danish1
deu8	German2
deuw	German2
ell8	Greek1
ellw	Greek1
esp8	Spanish1
espw	Spanish1
fin8	Finnish1
finw	Finnish1
plk8	Polish2
plkw	Polish2
ptb8	Portuguese2
ptbw	Portuguese2
rus8	Cyrillic1

Locale	Collation
rusw	Cyrillic1
ruw8	Cyrillic2
zdsw	Japanese1
zduw	Japanese1
zip8	Portuguese2

Locales Whose Default Collation Remains Caché Standard

Locale	Collation
chtw	Traditional Chinese
enu8	English
enuw	English
fra8	French
fraw	French
heb8	Hebrew
hebw	Hebrew
ita8	Italian
itaw	Italian
jpnw	Japanese
jpuw	Japanese (UNIX)
jpww	Japanese UTF-8
korw	Korean
nld8	Dutch
nldw	Dutch
zdtw	Japanese (DTM-J)

7.1.19 Accessing the Online Documentation

On Windows, when trying to access the documentation via the Cube, the userid assigned for the attempt is “UnknownUser”. When installing Caché with a security level of Normal or Locked Down, this username only has **%DB_DocBook:R** permission.

This is insufficient to read the Caché class reference documentation. Access to the class reference documentation requires that the user attempting to read the class documentation be authenticated.

Running program examples in the online documentation requires **%DB_SAMPLES:W**. If UnknownUser lacks this permission, then the button labeled **Run It** will not appear in any of the executable program examples.

Defining one or more roles which have the necessary permissions and assigning it to UnknownUser will establish the prior behavior. Alternatively, you may edit the application definition of "/csp/docbook" to add the role(s) whenever it is run.

7.1.20 Upgrading from a Prior Release

This section covers issues related to upgrading an existing Caché system to version 5.1.

7.1.20.1 No Upgrade from Field Test Versions

Customers running on any Caché 4.1.x or 5.0.x version may upgrade to Caché 5.1 at installation.

CAUTION: InterSystems does not support an upgrade from any of the versions used for field test of Caché 5.1. This includes the version of Caché 5.1 distributed to selected customers at DevCon 2005.

7.1.20.2 Use of DDP

If you were running DDP on an earlier version of Caché, you must edit your configuration file to allocate the proper number of network slots. They are no longer calculated by default.

- In the [Net] section of the configuration file, set the value of maxdsmport to the number of ethernet cards used for DDP.
- In the [config] section of the file, change the fourth parameter of the LegacyNetConn from 0 to 1.

Note: DDP will not start if these changes are not present.

7.1.20.3 $SYSTEM.OBJ.UpgradeAll()

The change in the compiler version, the reorganization of globals and routines, and the changes in Caché classes may generate a bewildering swarm of errors if **$SYSTEM.OBJ.UpgradeAll()** is invoked without prior planning and preparation.

7.1.20.4 Synthesized Role: %LegacyUnknownUser

In order to mimic prior behavior, during upgrade to version 5.1 a default role is created. This role is named **%LegacyUnknownUser**. The idea is that after upgrade from 5.0 and earlier versions of Caché where advanced security was not implemented, it will be common for no users to be defined. In this case, all users will be logged in as **UnknownUser**. If **UnknownUser** has no access privileges, the customer's operations will not be accessible to existing users until the administrators configure the system.

The **%LegacyUnknownUser** role is granted Read/Write access to the resource created for each customer-defined database that exists at the time of the upgrade installation and the resources shown

```
Name:               %LegacyUnknownUser
Description:        Legacy Unidentified Users
Roles granted by this role:
                    <none>
Resources owned by this role:
                    Resource                      Permission
                    --------                      ----------
                    %System_CallOut               U
                    %Service_SQL                  U
                    %Service_Object               U
                    %Service_Console              U
                    %Service_CallIn               U
                    %Service_CacheDirect          U
                    %Development                  U
                    %DB_USER                      RW
                    %DB_SAMPLES                   RW
                    %DB_%DEFAULT                  RW
Users owning this role:
                    <none>
```

In addition, use access to the following service resources will be granted subject to the indicated conditions:

Service	Privilege	Condition
%Service_ComPort	**Use**	On Windows, if service is enabled
%Service_Console	**Use**	On Windows, if service is enabled
%Service_LAT	**Use**	On Windows, if service is enabled
%Service_Telnet	**Use**	On Windows, if service is enabled
%Service_Terminal	**Use**	On UNIX and OpenVMS

After the administrator has configured the system appropriately, the **UnknownUser** user can either be disabled or the resources assigned to the role**%LegacyUnknownUser** can be gradually reduced via ^SECURITY or the System Management Portal as additional aspects of the application environment are brought under the control of Caché Advanced Security. This reduction of the privileges of the

%LegacyUnknownUser role or its removal is a manual step in the transition. It is not done automatically by Caché.

7.1.20.5 %LegacyCD and %LegacySQL

These are applied automatically to existing users only during upgrades to ensure that those users continue to have the same level of access in 5.1 that they did previously. New users aren't required to have these roles specifically.

7.1.20.6 Allow %-Global Access as in Previous Versions

The value of *Security.System.PercentGlobalWrite* is set true for upgrades. (For new installations it is set false.) This makes access to %-globals consistent with earlier versions. The value can be changed via the **^SECURITY** routine.

7.1.20.7 All Members of a Cluster Must Run the Same Caché Version

All members of an ECP cluster must be running the same version of Caché. If you upgrade one, you must upgrade all the rest.

7.1.20.8 Removal of CSP Gateway On OpenVMS Upgrade

The CSPGateway is no longer supported on OpenVMS. The material referencing it has been removed from the OpenVMS installation script and the code is no longer part of the product for OpenVMS.

Note: Any existing CSP Gateway files will be removed during a upgrade from a previous version.

7.1.20.9 Removal of Global & Package Mappings

During an upgrade from an earlier version, the following mappings to globals will be removed:

- all globals whose names begin with "^odd"
- ^rINDEXCLASS, ^rOBJ, ^ROUTINE
- ^mdd

Packages whose names start with "%Z" , "%z" , "Z" and "z" will have their definitions retained ("^oddDEF"), but will have their compiled class information removed. The "^odd" globals will be recreated when the classes are recompiled via **$SYSTEM.OBJ.Upgrade()**.

In addition, ALL class mappings that were defined in the configuration file (.cpf) will be discarded.

Note: If access to these globals is required, the administrator must manually construct the required mapping; they cannot be automatically converted. This will be the case if, for example, the system had defined global mappings so that multiple namespaces could share the same class definitions.

7.1.20.10 Trusted Application Definitions Removed

The Caché 5.1 security model does not support trusted applications. Anytime a user connects (or reconnects), he or she is prompted for a password if password authentication is turned on. If this is not what is desired, the administrator should turn authentication off for the Cache direct service.

Note: Any 5.0.x trusted application definitions are thrown away during a 5.1 upgrade.

7.1.20.11 Windows Network Server Username and Password Change

In previous versions, Caché would install and start its services on Windows with a default username of "_SYSTEM", and a password of "_sys" unless configured otherwise. The values for the username and password were set via the Configuration Manager using the **Advanced** tab and the "Input/Output" option.

In version 5.1, during upgrade, these values (or the default values if none were set) will be stored in the Windows service definition created for the upgrade.

Administrators may change the values using the Windows Management Interface. From the Windows **Start** menu, navigate to **Programs**, then **Administrative Tools**, and finally **Component Services**. Select the appropriate Caché service and from the **Action** menu choose **Properties**. The username and password are accessed via the **Log On** tab.

7.1.20.12 Global Replication Removed

Global replication is no longer supported in Caché 5.1. If found, the upgrade process will remove its use from the system and note this fact in the console log. The capability formerly provided by this can now be achieved by the use of shadowing. Please consult the high availability section of the system administrator book for details.

7.1.21 Java and Kerberos

Before using Java on Caché with Kerberos, you must edit certain configuration files, among them krb5.conf. Parameters in this file are set by running

```
java com.intersys.jgss.Configure
```

and responding to the prompts. On Windows, Solaris, and Linux, if krb5.conf is not found in the default location, Configure will search for it in the following locations:

- Windows

  ```
  c:\winnt\krb5.ini
  ```

- Solaris

  ```
  /etc/krb5/krb5.conf
  ```

- Linux

  ```
  /etc/krb5.conf
  ```

to obtain any template file information to be used when the file is created in the default location.

7.1.22 Recommendations

InterSystems has several recommendations to administrators setting up a Caché 5.1 system.

7.1.22.1 Enterprise Cache Protocol (ECP)

InterSystems strongly recommends the use of ECP for distrubuted systems. ECP represents a significant advance over predecessor networking approaches such as DCP. Customers currently using DCP will see improvements in performance, reliability, availability, and error recovery by converting to ECP.

7.1.22.2 Change Default Password Setting

When installing Caché with a security setting of "Minimal", the default passwords of all users created are set to "SYS". InterSystems suggests strongly that the password of these users be set to a different value as soon as possible so that, even though the security level of the system is low, control over access is established from the start of operation.

7.1.22.3 CACHELIB as Read-Only Database

In Caché version 5.1, for security reasons InterSystems has made CACHELIB a read-only database. This is a change from the previous practice. InterSystems strongly recommends that sites maintain CACHELIB as a read-only database. Those site- and application-defined globals, classes, tables and so on which might previously have been placed in CACHELIB should be moved elsewhere.

7.1.23 Limitations

The following limitations apply to Caché 5.1 or specific operating systems running Caché:

7.1.23.1 Maintaining Information Coherence Across Systems

On clustered Caché systems, it is highly desirable to maintain the same list of users, roles, applications, and so on across all the systems of the cluster. The initial release of Caché version 5.1 does not provide facilities for propagating changes in one system to others. This must be addressed by assuring that administrators manually make the same changes on each system.

7.1.23.2 Maintaining Coherence with Kerberos

Sites using Kerberos as the authentication mechanism must manually propagate changes to the list of valid users held by Kerberos to Caché. Future versions of Caché may provide mechanisms for doing this automatically but the initial release does not.

7.1.23.3 Consolidating Audit Logs

If a site wishes to run audit reports, or other analyses of their devising, on the audit data from several systems (for example, all the systems of a cluster), the individual audit logs must be consolidated manually.

7.1.23.4 Write Image Journal Files

The format of the WIJ (write-image journal) file has changed for Caché 5.1 to improve recovery in clustered systems. This has two consequences:

1. If an unresolved failure remains on any system to be upgraded, be sure to restart Caché and do a recovery before you upgrade to the new version.

   If you do not, the journal purge utility will not recognize journal files in the old format and will complain that there are corrupt journal files. To avoid this error, move the old journal files to a backup directory using the appropriate operating system commands before beginning the upgrade.

2. All members of an ECP configuration must be running the same version of Caché. If you upgrade one, you must upgrade the rest as well.

If you need to restore an older journal file to Caché 5.1, you can use the **JConvert** and **%JRead** routines.

7.1.23.5 Shadowing

A Caché upgrade converts existing *compatible mode* shadows to *fast mode*. The converted *fast mode* shadows may or may not work with the sources, depending on the source configuration. Caché 5.1 automatically performs endian conversion for *fast mode* shadowing.

Compatible mode (previously called *record mode*) is not supported.

Important: Shadowing in Caché 5.1 is not compatible with any prior release of Caché. Both the source server and destination shadow must be running on Caché 5.1.

7.1.23.6 Clusters

Caché 4.1, Caché 5.0, and Caché 5.1 clusters can coexist on the same hardware, but they cannot cluster together. If these clusters need to communicate with each other they need to use DCP, or preferably, ECP.

7.1.23.7 Management of Non-% Variables in Embedded SQL

Any non-% variables used by embedded SQL statements within a Caché ObjectScript procedure need to be added to the procedure's public variable list and be **New**ed within the procedure. While this is still a limitation in Caché, a change has been made to the macro preprocessor to make it easier to manually add these variables to the public and new lists.

When the **[Home] > [Configuration] > [Advanced Settings]** SQL setting **Retain SQL Statement as Comments in .INT Code** is "Yes" , along with the SQL statement in the comment, the non-% variables used by the SQL statement are listed in the comment text. This variable listing makes it easier to identify and cut and paste the variable lists into the MAC code public list and a new list.

7.1.23.8 Unicode in Global Names

Support for Unicode in global names is not yet fully operational and should be avoided.

7.1.23.9 Caché RPM kits

The Caché RPM kit installs into /usr/cachekit/5.1. Your /usr directory may be mounted read-only or may contain little free space, so you may want to change the location.

7.1.23.10 Database Interoperability

Databases created on earlier versions of Caché can be mounted on version 5.1 and, once they are upgraded, can be used there. But this process is not reversible. Upgraded databases cannot be moved back to earlier versions.

Note: InterSystems advises users who wish to move data bi-directionally between systems running version 5.0 and 5.1 to use the **%GOF/%GIF** routines to move data between the versions.

7.1.23.11 Upgrade Only Processes Local Databases

In Caché 5.1, **$SYSTEM.OBJ.UpgradeAll()** only scans local for local databases to upgrade. It ignores remotely mounted databases. These must be upgraded by running **UpgradeAll()** on the remote systems.

7.1.23.12 Caché Versions for ECP

Because of the upgrade to the compiler, systems in an ECP configuration must either be:

- all on version 5.1, or
- the data server must be on version 5.1 and the application servers can be either on version 5.0 or version 5.1.

Note: It is possible to run version 5.1 application servers with version 5.0 data servers, but this requires that the routines used by the application servers be mapped to databases local to the application servers. If you believe you have need to do this, please contact the InterSystems Worldwide Response Center for assistance.

7.1.23.13 Moving Applications from 5.1 to Earlier Versions

Porting an application from Caché 5.1 to an earlier release is problematic and depends on what features in this version the application depends on (compiler behavior, new streams implementation, changes in exported XML for applications, new class representations — to name a few). If you believe you have need to do this, please contact the InterSystems Worldwide Response Center for assistance.

7.1.23.14 ODBC and JDBC Compatibility

Due to a change in protocol, the ODBC and JDBC clients supplied with Caché 5.1 are compatible only with Caché servers from version 5.0.13 and later. Attempts to use connections to Caché servers in versions before 5.0.13 will result in errors.

7.1.23.15 RoseLink

RoseLink currently attempts to access Caché using only the standard SQL username and password. Therefore, it will not be supported on systems whose default installation security level is Normal or Locked Down. This restriction will be lifted in the next maintenance version of Caché 5.1.

In addition, the user must have **`%Development:Use`** permission in order to access classes for its use.

7.1.23.16 Dreamweaver

The connection that Dreamweaver MX uses to access Caché is not available with this version. This restriction will be lifted in a future maintenance release of Caché 5.1.

7.1.23.17 Perl and Python Language Bindings

The Perl and Python language bindings are supported only on 32-bit versions of Windows.

7.1.23.18 C++ Language Binding

The C++ language binding is supported only on the Windows platform using Visual Studio 7.1.

7.1.24 Platform-Specific Items

This appendix holds items of interest to users of specific platforms.

7.1.24.1 Windows

Help Format Change

The usage information for the commands **css.exe** and **ccontrol.exe** is now provided in HTML. Executing either command with a first argument of “help” will now invoke the default browser to display the help file.

New Japanese Locale

There is now a new Japanese locale for Windows (jpww). It is like the standard jpnw locale except that the default for Telnet Terminals is UTF8 instead of SJIS. This new locale is now installed by default for new Japanese installs on Windows. Upgrades to Caché maintain the previous locale.

Changes to %path% Environment Variable

To improve system security and the uniformity of locating Caché components, version 5.1 adds the file system directory name

```
\Program Files\Common Files\InterSystems\Cache
```

to the %path% system environment variable on Windows systems. It is added to the HKEY_LOCAL_MACHINE hive so that it applies to all users of this machine.

Visual Studio 7.1

On Microsoft Windows, Caché is now compiled with Visual Studio 7.1. User applications communicating with Caché (for example, those using the CALLIN or CALLOUT interfaces) must be upgraded to this version of Visual Studio.

7.1.24.2 Windows XP Professional

Microsoft Mapped Network Drives

Mapped drives are not supported on Windows XP Professional — Due to security improvements in Windows XP Professional, Microsoft discourages users from using mapped drives; using them results in different behavior than in the past.

We recommend that XP Professional users follow these procedures to access mapped drives from the GUI tools or from telnet sessions:

- For remote mapped drives, enter the user name and password in your configuration as before. In addition, edit the **ZSTU** startup routine and add this line for each drive you have mapped.

  ```
  Set x=$zf(-1,"net use z: \\someshare")
  ```

- For virtually mapped drives, add this line for each drive mapped with the **subst** command:

  ```
  Set x=$zf(-1,"subst q: c:\somedir\someotherdir")
  ```

You cannot add more mappings after startup.

Important: The above procedure is meant for development situations where only one user is expected to log on to Windows, and the user name entered in your configuration is the same user. In any other situation, such as a Terminal Server environment, the results are unpredictable.

The following notice from Microsoft refers to this problem:

[Redirected Drives on Windows XP Professional: On Windows XP Professional, drive letters are not global to the system. Each logon session receives its own set of drive letters A-Z. Thus, redirected drives cannot be shared between processes running under different user accounts. Moreover, a service (or any process running within its own logon session) cannot access the drive letters established within a different logon session.]

Another approach to using the mapped drives is to start Caché like this:

```
\cachesys\bin\ccontrol start configname
```

With this approach you do not have to add anything to the **ZSTU** routine, and you do not have to enter a user name and password. In addition, drives you map or map with a path using the **subst** command after startup are available. The limitation of this approach is that Caché only runs as long as the user that starts Caché stays logged on.

7.1.24.3 Windows Enterprise Server 2003

Internet Explorer

The version of Internet Explorer distributed with this version of Windows has every security related configuration option disabled. The result is that various pages displayed by the System Management Portal are affected; for example, information generated by scripts will not materialize because the scripts will not be run. The proper behavior can be restored by changing the internet security level setting from "High" to "Medium".

The first time a user accesses the System Management Portal on a particular system, Internet Explorer will prompt to ask if this site should be added to the "trusted" list. Answering in the affirmative, will also change the internet security level for that site to Medium.

7.1.24.4 Mac

Support for Xalan

Support for Xalan, an XSLT (Extensible Stylesheet Language Transformation) processor, is only available on OS x 10.4.

7.1.24.5 OpenVMS

ECO Required for Access Using Kerberos on Itanium

Applications attempting to access OpenVMS servers that use Kerberos authentication must install the patch, HP-I64VMS-TCPIP-V0505-11ECO1-1, available at the `ftp://ftp.itrc.hp.com/openvms_patches/layered_products/i64/` ftp site. The ECO is for TCP/IP, not the actual operating system. Without this patch, the server will often transmit erroneous response packets back to clients using the C++ binding, ODBC, JDBC, and Studio.

Note: This ECO applies only to OpenVMS on Itanium hardware. It is not needed for OpenVMS on Alpha.

CSP Gateway Removed

Support for the CSP Gateway on OpenVMS has been removed. Please refer to the OpenVMS installation instructions for more details.

Password Masking Limitation in GSS

When attempting to access Caché via JDBC on systems using Kerberos, if no credentials for the user are found, and the identity of the user is not supplied by the caller, JDBC will ask Kerberos to authenticate the caller. When this happens, due to the characteristics of terminal IO on OpenVMS, echoing of the password will neither be suppressed nor masked.

Using the SOAP Client Wizard from Studio

An attempt to start the SOAP wizard from Studio will fail unless the application, **`/isc/studio/template`**, is set up to point to the current web server used for OpenVMS.

Caché Processes and /SYSTEM

All processes that are part of Caché ru with UIC=[1,4]. Therefore, all Caché-related logical devices used by these processes, for example, those mentioned in the .cpf file, must be defined in the system table (defined with /SYSTEM) to avoid access errors.

WebServerName and WebServerPort

In version 5.1, Studio is unable to access the Class Documentation unless both the WebServerName and the WebServerPort are defined. These are found in the Miscellaneous category of the System Management page, **[Home] > [Configuration] > [Advanced Settings]**.

7.1.24.6 AIX

IBM Java Runtime and Kerberos

On systems using the IBM Java runtime environment (AIX 32–bit, 64–bit and SUSE Linux Enterprise Server), use of **kinit** is not compatible with Kerberos principal name and password prompting, or the principal name and password API. To use **kinit**, change the file

```
${java.home}/lib/security/iscLogin.conf
```

so that the module, com.sun.security.jgss.initiate, has the option

```
useDefaultCcache=true
```

With this runtime, only the Java routine at

```
{java.home}/bin/kinit
```

works and not the native Kerberos routine at

```
/usr/krb5/bin/kinit
```

NFS-Mounted Filesystems And Exclusivity

Cache uses O_EXCL (exclusive access) when creating Caché database (.dat and .ext) and lock (.lck) files. However, it is a known limitation that NFS does not guarantee this exclusivity.

7.1.24.7 Linux

Netapp NFS-Mounted FileSystems

On Netapp NFS-mounted filesystems under Linux, a file created by a suid:sgid executable has different, non-UNIX standard, owners than on standard filesystems. The sgid bit on the executable fails to take effect, while the suid bit succeeds in setting the owner of the file to the owner of the executable. This behavior has been observed only on Netapp systems.

7.1.24.8 Red Hat 3.0 / 4.0 And IBM WebSphere MQ

If you plan to use the MQ interface, IBM WebSphere MQ version 6.0 is required when running Caché 5.1 on Red Hat version 3.0 and 4.0.

7.1.24.9 Linux / AMD64

New License Required

When upgrading from Caché from a Linux implementation on an Intel processor to Linux on AMD64, a new Caché license is required. As noted on the InterSystems Web site:

“Because of the significant differences between 32-bit and 64-bit CPUs, InterSystems delivers different Caché software for them and, consequently, they are different platforms for licensing purposes. As a result, Platform Specific Caché licenses cannot be transferred from one to the other. (Normal trade-in policies apply.) Platform Independent licenses can, of course, be transferred at no charge.”

7.1.24.10 SUSE Linux Enterprise Server

IBM Java Runtime And Kerberos

On systems using the IBM Java runtime environment (AIX 32–bit, 64–bit and SUSE Linux Enterprise Server), a different Kerberos **kinit** is needed. See the description includes with AIX.

Caché Terminal With Kerberos Authentication

SUSE Linux Enterprise Server 9 running on AMD64 handles packages slightly differently from other versions of Linux. A result of this is that attempting to use Caché terminal on this system with Kerberos authentication may encounter errors in the case where the installer has not chosen to install the developer packages. In this instance, the following packages must be installed to ensure proper operation:

- heimdal-devel
- heimdal-devel-32bit

The packages are most easily located by using the search facility to locate all the packages whose name begins with "heimdal". In most installations (except "full"), the list will show the two packages named above as unselected. Select them and continue with the installation.

7.1.24.11 UNIX

Installing Caché On UNIX

Users may install Caché on UNIX so that cachesys is not the default directory. The directory path is assumed to be in an environment variable, *CACHESYS*. The **ccontrol** and **csession** commands use this environment variable. If it is defined at installation time, Caché is installed in that directory. If it is not defined, Caché is installed in the standard UNIX location, /usr/local/etc/cachesys.

Both **ccontrol** and **csession** expect to find the registry in the same directory where their executable was found. For security reasons, **ccontrol** verifies that the protection on the registry is root as the owner and writable only by root.

7.1.24.12 Tru64 UNIX

Setting File And Directory Ownership

For Tru64 systems, unlike other UNIX file systems, group ownership does not come from the group id of the creating process. Instead, the group ID of the file is set to the group ID of its parent directory.

However, when the vfs subsystem attribute "sys_v_mode" is set to 1, the group ID of the file is set either to the group ID of the process or, if the S_ISGID bit of the parent directory is set, to the group ID of the parent directory If the group ID of the new file does not match the effective group of the process or one of its supplementary group IDs, the S_ISGID bit of the new file is cleared.

In general, this will present no problems since the groupid of all directories created by Caché utilities is properly set to the correct group owner. But there are circumstances which can cause problems. For example, if an administrator uses ^DATABASE to create a database in an nonexistent directory, ^DATABASE will create the directory, but it does not adjust the groupid of the newly-created directory, which is inherited from the parent directory. As a result, the database, with its groupid inherited from the directory, may be inaccessible to cacheusr. Other Cache utilities (e.g., journal and shadow) that create directories have the same problem.

Note: It is recommended that System Administrators set the sys_v_mode to 1 on all file systems and directories used by Caché to ensure smooth functioning of the system. For further information, please refer to the manpages for the **open(2)** system call.

7.1.24.13 HP-UX

Entropy Source Component

The Caché cryptographic random number generator (use, for example, to encrypt and decrypt databases) requires a source of true randomness (entropy) in order to initialize its internal state. All supported UNIX platforms except HP-UX 11i provide the special device file, /dev/urandom, that provides a source of true entropy based on kernel thread timings. On HP-UX, this functionality is part of the HP-UX Strong Random Number Generator available as a free, optional component supplied and supported by HP.

CAUTION: If this component is not installed, Caché uses other sources of entropy available on the system. However, these have not been analyzed for randomness, and therefore the encrypted values generated by Caché are not as strong as they could be otherwise.

7.1.24.14 Solaris

Applications running on Solaris will fail to obtain an initial set of credentials when using a password. This happens, for example, when trying to access a Caché instance requiring Kerberos authentication via TERMINAL. Sites intending to use Kerberos authentication with Caché will require patches to Solaris, namely,

- For Solaris 10, 121239–01 and 120469–03 (or greater).
- For Solaris 9, 112908–22 and 112907–06 (or greater).

7.2 Developers

This section contains information of interest to those who have designed, developed and maintained applications running on prior versions of Caché. Although InterSystems placed great importance on upward compatibility in version 5.1, the increased emphasis on security resulted in the redesign and re-implementation of some core parts of Caché. The effects of this necessarily affect existing applications.

The items listed here are brief descriptions. In most cases, more complete descriptions are available elsewhere in the documentation.

7.2.1 System Management Portal

Prior to this version, how Caché was administered depended heavily on the platform where Caché ran. With version 5.1, InterSystems introduces a new administrative interface that is common across all platforms. Caché 5.1 now uses a browser-based interface, the System Management Portal, for system management.

Although mainly for administrators and operators, developers may occasionally need to use some of its functions. A brief summary can be found in the Administrator section of this document and more complete information on the System Management Portal can be found in the System Administration documentation.

7.2.2 Privileged Operation

Caché has always had the concept of "privileged" operations. In Caché 5.1, this concept has been more clearly defined, strengthened and made more granular. Commands, routines, functions, methods and so on that are privileged must meet one of two criteria before Caché will allow them to proceed:

1. They must be invoked by a unmodified routine that is loaded from the CACHESYS database.
2. They are invoked by a user who holds a role granting permission to perform the operation. In most cases, privileged operations require **`%DB_CACHESYS:W`**, but certain operations may deviate from this.

It either of these conditions is true, then the requested operation will proceed.

7.2.3 Recompile User Applications After Upgrade

User application classes, and routines containing embedded SQL statements must be recompiled after upgrading to this version of Caché as noted in the Administrator section of this document.

7.2.4 CACHESYS and CACHELIB Reorganized

Any robust security implementation shares a number of characteristics with other like systems. For example:

- The number of functions implementing the security module should be kept as small as necessary.
- These should be collected together and isolated from other system functions so they can be protected.
- They must be independently verified for correct operation and benign failure.

As part of the effort to add increase security in Caché, InterSystems has reviewed the low-level routines present in the CACHESYS and CACHELIB databases in light of these requirements. The result is that

the contents of these database have been reorganized. CACHESYS (the manager's database) now contains only the low-level routines necessary for system management. Everything else has been moved to CACHELIB.

The following are brief guidelines to the changes.

For System classes:

- Methods in classes of the %SYS package:
  - These reside in the manager database (CACHESYS) because they invoked protected system routines.
  - Since their names start with “%”, they are mapped into all other namespaces.
- Method of classes in the %System package:
  - These reside in the CACHELIB database which is mounted read-only by default. They do not invoke protected functionality, but reside there to support legacy applications.
  - Since their names start with “%”, they are mapped into all other namespaces.
- Methods in the Sys and System packages reside where the %Sys and %System packages reside, respectively. However, because their names do not start with “%” they are visible only within those databases.

For the system functions whose name is of the form, **$SYSTEM.<name>**:

- If the name is one associated with a method in an internally known system class, it invokes that method.
- Otherwise, it attempts to invoke the %System method by that **<name>**. So **$SYSTEM.SomeClass.That Method()** is equivalent to **##class(%System.SomeClass).ThatMethod()**.

And finally, for globals:

- All globals whose names start with “%q” are mapped to CACHELIB (default read-only).
- All other globals map to CACHESYS (default read-write).

The mappings can be displayed in more detail using **^%SYS.GXLINFO**.

7.2.4.1 CACHELIB Is Mounted As Read-Only

As part of the reorganization of CACHESYS and CACHELIB, all of the information that is modifiable during normal operation has been collected into CACHESYS. Therefore, CACHELIB is mounted as a read-only database by default.

7.2.4.2 Access To %-Globals Is More Restrictive

By default, routines do not have write permission on %-globals that reside in other databases. In version 5.1, these rules are now consistently enforced. This can be changed via the System Management Portal at **[Home] > [Security Management] > [System Security Settings]** by changing the setting for "Enable writing to %-globals" to "Yes".

7.2.4.3 Permissions On CNLS

The CNLS application is used to change the locale of a Caché installation. Running it now requires **`%Admin_Manage:U`**.

7.2.4.4 Authenticated Namespace And Routine Override Command Line

In prior versions, when a namespace and routine were supplied as a parameter on a command line, the process created would always use that namespace, and would override any namespace or routine specified by the security mechanisms of that version.

In version 5.1, if Caché is installed with the MINIMAL setting, csession will work as before. If the user needs to be authenticated, the namespace and routine for that user will override any namespace or routine setting supplied on the command line.

7.2.5 Changes To Routines

7.2.5.1 Routines And Globals Moved

In Caché 5.1, all %-routines and %-globals were reorganized as noted above.

Note: If there are user- or site-supplied routines whose names begin with "%", they must obey these same rules. These changes require administrative privilege because, by default, the CACHELIB database is set read-only at installation time and cannot be altered.

Unless routines added at the site need to create globals in CACHELIB during normal operation, InterSystems recommends that, after installing these routines, CACHELIB be made read-only once again.

7.2.5.2 Routines Renamed By Removing "%"

The review of Caché system functions resulted in a number of routines being designated as system management functions whose use needed to be controlled. Therefore, the following routines have been renamed by removing the "%" from their name, thus placing them within the protection of the manager's database:

- **BUTTONS**
- **COLLATE**

- **DMREPAIR**, **DSET**
- **LANG***
- **MATH**
- **NLS**, **NLSCOMP**, **NLSLOAD**, **NOREP**
- **ROLLBACK***
- **SS**, **SSVNJOB**, **SSVNLOCK**, **SSVNROUTINE**, **ST**
- **UPDATECLASS**
- **Wcomm**, **Wpfiles**, **Wr**, **Wr1**, **Wsback**, **Wsdb**, **Wsdba**, **Wsmkey**, **Wsnls**, **Wsnls2**

Important: This change means that these routines must be invoked from a non-edited routine in the CACHESYS database and not be part of any indirection; or else be in a process that holds WRITE permission on the CACHESYS database.

7.2.5.3 Routines Renamed

To further emphasize their relationship to system function, some routines were renamed:

Previous Name	New Name
%DM	%SYS.DATABASE
%FILE	%SYS.FILE
%GCREATE	%SYS.GCREATE
%GD	%SYS.GD
%GIFMSM	%SYS.GIFMSM
%GLO	%SYS.GLO
%GOF1	%SYS.GOF1
%GSETNS	%SYS.GSETNS
%GSET	%SYS.GSET
%GXLINF1	%SYS.GXLINF1
%GXLINFO	%SYS.GXLINFO

Previous Name	New Name
%LICENSE	**%SYS.LICENSE**
%MACHINE	**%SYS.MACHINE**
%MONLBL	**%SYS.MONLBL**
%NOJRN	**%SYS.NOJRN**
%PMODE	**%SYS.PMODE**
%RI2	**%SYS.RI2**
%RIMF	**%SYS.RIMF**
%RI	**%SYS.RI**
%SYSCONV	**%SYS.SYSCONV**
%Wcdu	**%SYS.Wcdu**
%Wgr	**%SYS.Wgr**
%Wgs	**%SYS.Wgs**
%Wsys	**%SYS.Wsys**

7.2.5.4 Routines Eliminated

During the review, some routines were identified as duplicating functionality provides elsewhere. These were removed:

- **%CLI** — The same functionality is available from Caché through **$zf(-1)**. On UNIX, OpenVMS, and Mac, command line interpretation is done via **!<command>**. On Windows systems, the DOS command **START** performs this function.
- **%DKIOERROR** — Calls to it should be replaced with $$$ERROR or **$SYSTEM.Error** usage.
- **%GED** — Use **%GCHANGE** and %Library.Global methods instead.
- **%GDOLD** — This routine has been removed.
- **%GROWTH** — The functions of this routine have been moved to the SYS.Database class.
- **%GTARGET** — This routine has been removed.

- **%LM** — The functions of this routine have been included in the SYS.Lock class.
- **%LMFCLI** — The functions of this routine have been included in the $SYSTEM.License class.
- **%qserver** — The user accessible entrypoints have been moved into $SYSTEM.SQL.
- **%RMAC** — This routine has been removed.
- **%START** — This routine has been removed.
- **%USER** — This routine has been replaced by $USERNAME.
- **%UTIL** — This is an internal routine which has been removed. Its message logging function has been converted into a system macro, LOGMSG.

7.2.5.5 Stub Routines Added

Some frequently-invoked routines were moved to CACHESYS (**%DM**, **%LICENSE**, **%GD**, and **%SS**) and were renamed. Stub routines that call the new routines were left in their place as a compatibility aid. Applications are encouraged to move to using the new names.

In adding the stub routines, the tag, SYS, has been removed from the **%SYS** routine.

7.2.5.6 Routines Deleted

In addition to the changes noted above, internal and obsolete routines were removed from these libraries. If you suspect that this may be affecting your application, please contact the InterSystems Worldwide Response Center for assistance.

7.2.5.7 No Mapping For %LANG Routines

Caché 5.1 ignores any routine mappings for the %LANG* routines that are used to provide language extensions. The routines executed will always be those in he %SYS namespace (CACHELIB).

7.2.6 Class Changes

During the development of version 5.1, a number of changes were made to improve development accuracy and reduce ambiguity when using classes. They are collected in this section

7.2.6.1 Classes Replaced

The following classes have been removed from the system because they have been replaced with classes providing better functionality. They are listed below:

Class	Replacement

Class	Replacement
%CSP.Util.LoginForm	None. This is internal class, used only by other InterSystems-developed classes, has been superceded by the CSP login mechanism.
%CSP.Util.NamespaceForm %CSP.Util.Password	None. This is an internal class used only by other InterSystems-developed classes.
%Library.CacheProperty	None. This is an internal class used only by other InterSystems-developed classes.
%Library.StreamAdaptor	This class has been replaced by several classes providing specific capabilities for differing circumstances, for example, %Library.BinaryStream, %Library.CacheStream, %Library.CacheStreamLegacy, %Library.CharacterStream, %Library.FileBinaryStream, %Library.FileCharacterStream, %Library.SerialStream, %Library.Stream.
%Library.StringTimeStamp	See %Library.TimeStamp for replacement functionality.
%Studio,SourceControl.Example	See %Studio.SourceControl.Base
%SYSTEM.Database	The functions of this class have been replaced by those of SYS.Database.
%SYSTEM.ECP	The functions of this class have been replaced by those of Config.ECP and SYS.ECP.
%SYSTEM.Process	See SYS.Process.
%SYSTEM.Server	None. This is an internal class used only by other InterSystems-developed classes.

Class	Replacement
%SYSTEM.Monitor.*	The Monitor facilities were completely redesigned for version 5.2. Applications which formerly used the %SYSTEM.Monitor classes will have to be modified, perhaps extensively. Please refer to the %Monitor and %MonitorTools pacvkages and to the Config.Monitor and Config.MonitorTools classes.

7.2.6.2 New Classes

The following classes are new in this version of Caché:

- %Library.AbstractResultSet
- %Library.CacheCollection, %Library.CacheLiteral, %Library.CacheObject, %Library.CachePopulate, %Library.CacheString, %Library.Collate
- %Library.DataType, %Library.Device
- %Library.Function
- %Library.Global, %Library.GlobalEdit, %Library.GTWConnection, , %Library.GTWResultSet
- %Library.IJCDevice
- %Library.JavaDoc, %Library.JournalRecordType, %Library.JournalState
- %Library.ObjectJournal, %Library.ObjectJournalRecord, %Library.ObjectJournalTransaction
- %Library.PersistentProperty, %Library.Prompt
- %Library.RemoteResultSet, %Library.RowSQLQuery
- %Library.ShadowState, %Library.ShadowType, %Library.SwizzleObject
- %Library.Utility

Programmers who relied on an unqualified class name resolving to the correct location may discover that the new classes added to %Library now cause naming conflicts if the application defined classes with any of these names.

7.2.6.3 Name Case Conflict In Class Compilation

In version 5.0, Caché allows a class to define a method which has the same spelling as a method of its superclass, but differs in case. For example, a method called **%save()**, defined in class that inherits

from %Library.Persistent would be considered a different method from the **%Save()** method of the superclass.

In version 5.1, this situation produces a compilation error. For example, CSP applications that had defined methods of **include()** or **link()** will find that these are now in conflict with **%CSP.Page.Include()** and **%CSP.Page.Link()** respectively.

7.2.6.4 Ambiguity Resolution When Simple Class Names Used

If an application makes a reference to a class whose name begins with a percent-sign and which does not specify its package name, the class compiler looks for the class in the %Library package for the class. Thus,

```
Set BaseDir = ##CLASS(%File).ManagerDirectory()
```

is interpreted as if it had been written

```
Set BaseDir = ##CLASS(%Library.File).ManagerDirectory()
```

Programmers who relied on the unqualified class name resolving to the correct location will discover that the new classes added to %Library may now cause ambiguity in naming if the application defined classes with the same name, for example %Utility.

7.2.6.5 Class Member Naming Checks Made More Strict

In Caché version 4.1, you were not allowed to define two class members with the same name but different case. In version 5.0, however, a bug was introduced that failed to report these errors.

In Caché version 5.1, this bug is fixed but in order to allow these classes that did previously compile on 5.0 to still compile application developers can disable this check by setting the 'Strict Checking' flag is set to off. This is done by executing the command:

```
Set ^%qCacheObjectSys("strictchecking") = 0
```

Note: In order to set this flag, you will have to change the permission on CACHELIB from Read-Only to Read/Write. This is done using the Management Portal, **[Home] > [Configuration] > [Local Databases]**.

7.2.6.6 New Methods Added To %Library.Persistent

Several new methods are implemented in %Persistent and are inherited by all persistent classes. These methods are:

- **%LockId**: acquires a lock on an instance of a class
- **%UnlockId**: releases a previously acquired instance lock
- **%LockExtent**: acquires a lock on every instance in an extent

- **%UnlockExtent**: releases a lock ion extent
- **%GetLock**: attempts to get a lock on an instance, but will escalate the lock to the extent of which it is a member, if necessary

Note: Changes to the underlying storage mechanisms that affect persistent objects are detailed here.

7.2.6.7 Conflicts With User-Written %-Methods

User applications with method names that start with "%" should check to make sure that there are no conflicts with methods supplied by InterSystems in "base" classes such as %Library.Persistent. This version of Caché has significantly increased the number of such names.

7.2.6.8 IdKeys Now Have <Indexname>Exists and <Indexname>Open Methods

This version of Caché now supplies <indexname>Exists and <indexname>Open methods for IdKeys.

All persistent class have an IdKey. If one is not explicitly defined or inherited from a superclass, then an index named IdKey<n> will be generated where <n> is an integer that is appended to the root of "IdKey", if another index named IdKey already exists. This index is defined as a system generated index.

In prior versions, the index was generated during class compilation. No inheritance resolution was applied to the generated index, and no index methods were generated.

7.2.6.9 IsValidDT Changes

Method No Longer Generated By Default

In version 5.1, user-written datatype classes that extend the%Library.DataType class no longer have the **IsValidDT()** method automatically generated. The previous behavior can be restored by executing

```
Set ^SYS("ObjectCompiler","GenerateIsValidDT") = 1
```

Note: Each namespace where the older behavior is and then recompile all affected routines.

Method Return Type Changed

In previous versions, the class instance method, **%IsValidDT()**, returned a value of type %Integer. In Caché 5.1, it more correctly returns a %Boolean.

7.2.6.10 Methods Supporting SQLCOMPUTECODE

Caché allows classes to define SQL computed properties by declaring them with the attribute, SQL-COMPUTED, and providing SQL code to computed the desired value. The value can be transient, calculated or storable.

For computed properties a **<property>Get()** method is generated that invokes **<property>Compute()** as needed. SQLCOMPUTECODE allows for other values to be referenced during computation. These references are to SQL columns (preserved for backward compatibility) and are converted to property names during method generation.

If the SQL column references a column projected from an embedded class, then **<property>Compute()** will generate an extended reference to the embedded property.

Note: Using embedded properties in SQLCOMPUTE code breaks encapsulation. One problem with breaking encapsulation is with "triggered" computed fields, that is, when SQLCOMPUTEONCHANGE is declared. Embedded property references are not supported in SQLCOMPUTEONCHANGE.

7.2.6.11 Changes To Inheritance Processing

In previous versions, the inheritance rules for classes were not always as expected. For example, if a user created a class, User.MyClass, that was a subclass of %Library.Persistent, Caché would automatically inherit the default package name of the superclass, %Library, as a #import into User.MyClass. A consequence of this is that if User.MyClass contained a property declared such as

```
property A as String;
```

Caché would try resolve this by looking in both the User and %Library packages. If User had a class, User.String, Caché would report a classname conflict even though the user probably intended to reference User.String. The workaround was to fully qualify the property name as in

```
property A as User.String;
```

Caché 5.1 will still inherit any explicit #import settings from the superclasses, but it will not automatically add the superclass package names to the #import. So in the example given the *A* property would resolve to 'User.String' without any name conflict errors.

Caché still uses the current class packagename in resolving names; User.MyClass will still use 'User' as a #import for its name. But this is no longer true for subclasses.

More explicitly, Caché will always resolve the name in the context where it was first defined and not the current classname. For example, suppose User.MyClass defines a method, **X()**. If a class MyPackage.MyClass inherts from User.MyClass, when it is compiled Caché will compile the inherited **X()** method in MyPackage.MyClass but resolve any unqualified classnames used in this method in the context of User.MyClass because this is where **X()** was defined.

7.2.6.12 Stream Implementation Has Been Modified

In version 5.1, cloning a class containing a stream member works differently from earlier releases. What happens now is:

- If the stream member is a serial stream, the oref of the stream is copied to the clone.

- If the stream is an instance of the "older" stream implementations:
  - %Library.FileBinaryStream
  - %Library.FileCharacterStream
  - %Library.GlobalBinaryStream
  - %Library.GlobalCharacterStream

  the oref of the stream is copied to the clone.
- In all other cases, Caché will make a copy of the stream contents in a new stream of the same type and place the oref of the new stream into the clone.

If an application wishes to retain the oref of the original stream, it can do so with

```
Set ..ThatStream = oref.%ConstructClone(0)
```

7.2.6.13 XML Export Replaces CDL

In this version, CDL is no longer available as an export format for classes. Users should export their classes in XML instead. CDL will still be accepted as an import format for this release.

7.2.6.14 Persistent Superclasses Must Reside Outside of CACHELIB

Subclasses of persistent classes currently store some of their extent information with the extent information of their superclass. Because CACHELIB in Caché version 5.1 is now a read-only database, it is no longer possible to subclass persistent classes residing in CACHELIB by default. Attempting to do so will result in a <PROTECT> error. This is true even if the persistent classes were created locally and stored in CACHELIB.

The only exception to this is classes which are marked as SERIAL. They do not have extent information since their instances are embedded in the class that references them.

7.2.6.15 TRUNCATE Default Changed For %Library.String

Strings have, among their other parameters, settings for MAXLEN and TRUNCATE. The value of MAXLEN specifies the maximum permissible length of the string. The value of TRUNCATE specifies how to enforce the maximum length limit.

- If TRUNCATE is set to true, Caché will store only the first MAXLEN characters in a variable declared as type %Library.String ignoring the rest of the string.
- If TRUNCATE is set to false, an attempt to assign more than MAXLEN characters to the variable will return an error status.

In Caché version 5.1, the default value of TRUNCATE for new instances of %Library.String will be false. In previous versions it had been true. Note that this applies only to new strings created in version 5.1. Older items of type string will still have the defaults from the time they were created.

7.2.6.16 Support For Legacy %Close() Behavior Dropped

In version 5.0, Caché changed how it handled objects that were closed. The object was destroyed upon **%Close** if its reference count went to 0. The OREF associated with the object would be removed once it was marked "inactive" ; that is, all references to it were gone.

When this behavior was introduced, it was possible to have Caché use "legacy support" for **%Close** instead — the method used in versions prior to 5.0 — via the call

```
Do $ZU(68,56,1)
```

In this mode, Caché decrements an object's object-level reference count upon **%Close()** and removes it from memory when the count reaches 0. No provision was made to prevent re-use of the OREF.

In Cache 5.1, legacy mode has been removed. Calling this function will result in a <FUNCTION> error.

7.2.6.17 %DeleteExtent() Behavior Improved

In prior versions, the **%DeleteExtent()** method always returned $$$OK, even if not all instances in the extent were deleted. In version 5.1, its behavior now better matches expectations; it only returns $$$OK if all instances of the extent were successfully deleted.

7.2.6.18 Method Compilation And Return Values

In previous versions, if a method was declared to return a value, the method compiler would insert a

```
Quit ""
```

if the last line of the method did not begin with a **Quit** command. This approach, however, hid subtle programming bugs because the method the developer wrote did not, in fact, return a value when it was supposed to.

In version 5.1, this is no longer done. The method compiler will only insert a simple

```
Quit
```

instead if the last line of the method does not contain one. Thus, invoking a method (function) that does not return a value when it is declared to will result in a <COMMAND> error.

7.2.6.19 Required Relationship Collections Cannot Be Empty

If an application specifies that a "child" or "many" side of a relationship collection is required, Caché now make sure this contains at least one element. If the relationship is empty at the time the instance is saved, Caché reports an error on %Save, for example:

```
ERROR #5662: Relationship child/many property 'Sample.Company::Employees
(1@Sample.Company,ID=)' is required so must have at least one member
```

7.2.6.20 Cycle Checking For XML Exports

In Caché 5.1 XML-enabled classes check their hierarchy before export to determine if there is a cycle present. This check is on by default, but may be disabled by appending ",nocyclecheck" to the Format property of %XML.Writer.

Note: If this check is disabled, and a cycle is present, a <FRAMESTACK> error will result.

7.2.6.21 Task Changes

Task Manager Hierarchy Upgraded

The Task Manager now uses the Caché class hierarchy more completely. All user tasks must now subclass the class, %SYS.Task.Definition.

Subclasses can thereby introduce additional properties which will be available during the task execution. The user interface will then interrogate the definition to request the property values from the user. For example,

```
Class %SYS.Task.IntegrityCheck Extends %SYS.Task.Definition
{
Property Directory As %String [ InitialExpression = {$zu(12)} ];
Property Filename As %String [ InitialExpression = "INTEGRIT.LOG" ];

ClassMethod DirectoryIsValid(Directory As %String) As %Status
{
  If '##class(%Library.File).DirectoryExists(Directory)
  {
    Quit $$$ERROR($$$GeneralError,"Directory does not exist")
  }
  Quit $$$OK
}

/// This method is responsible for executing the task
/// At the scheduled time, the Task Manager
/// - creates an instance of this object,
/// - sets any property values using the stored "settings" for the task,
/// - and invokes this method to execute the task.
/// In order to execute a real task, override this method in a subclass.
Method OnTask() As %Status
{
  Do Silent^Integrity(..Directory_..Filename)
  Quit $$$OK
}

}
```

ContinueAfterError Property Removed

The ContinueAfterError property in the class %SYS.TaskSuper has been removed because it is too general. Tasks which depended on it must be redesigned to handle their own error conditions.

Other Task Improvements

In addition, the following improvements have been made to task management in Caché 5.1:

- There is a RunLegacyTask class that provides the **ExecuteCode()** method for compatibility with earlier versions.
- If a running task encounters any kind of error, it is suspended.
- The methods, **RunOnce()** and **RunNow()**, will now make a suspended task active.
- Task startup is prevented if there is no license found for this instance of Caché.

7.2.7 CDL Support Dropped In Following Releases In Favor Of XML

In Caché 5.1, CDL was removed as an option for exporting classes (see XML Export Replaces CDL) in favor of the industry-standard XML. Caché 2008.1 will complete this transition. CDL will no longer be available as a format for import to Caché, either as import or output.

Furthermore, for new platforms added in 2007.1 that are not newer versions of existing platforms, InterSystems may decline to provide support for CDL at all. For the exact details on each Caché platform, please refer to the Supported Platforms documentation.

For those customers that have program archives in CDL format, InterSystems recommends importing them into Caché 2007.1, and exporting them as XML.

7.2.8 SQL Differences

In the transition to version 5.1, the following changes were made in SQL that may affect existing programs.

7.2.8.1 Caché And SQL Users Unified

In prior versions, the list of valid Caché user names and the list of valid SQL user names were unrelated and were governed by different security mechanisms. In version 5.1, this is no longer true. All SQL user names are Caché user names, and vice versa. What each is permitted to do is determined by the same security mechanism.

7.2.8.2 Atomic SQL statements

In Version 5.1, the SQL statements DELETE, UPDATE, and INSERT...SELECT have been made atomic. That is, the statement either completes successfully or no rows in the table are modified.

In previous versions, it was the responsibility of the application to detect an incomplete operation and roll back the update (if desired). Now, if any row fails to update, none of the rows in the table will be updated by the statement.

7.2.8.3 SQL Passwords Are Case-Sensitive

In Version 5.1, for security reasons, SQL uses the same password mechanisms as Caché. One consequence of this is that SQL passwords are now case-sensitive. Previously, they were not.

7.2.8.4 Table Ownership Interaction With $USERNAME

This means that tables created through the use of DDL will have as owner the value of $USERNAME at the time they were created. When creating a class or table by any other means, the class's OWNER keyword is not defined unless the developer explicitly defines it. When a class is compiled that projects a table and the class's OWNER keyword is NULL, the table's owner is set to _SYSTEM.

This interpretation is the same as in previous versions. What has changed is that there is no default to an OWNER of _SYSTEM when creating tables through DDL in 5.1.

7.2.8.5 Delimited Identifiers Are The Default

Caché version 5.1 installs with the value for “Support Delimited Identifiers” as true. This means that a double-quoted string (“My String”) is considered a delimited identifier within an SQL statement. Prior versions of Caché had this parameter set to false: a double-quoted string was treated as a string constant or literal string. The value of this parameter can be changed via the System Management Portal at **[Home] > [Configuration] > [Advanced Settings]**.

7.2.8.6 %msql Eliminated

This variable was used in Caché ObjectScript to specify a valid user name for SQL access from embedded SQL. A valid user name was one that was registered in the User Table.

In Caché 5.1, the SQL username is now extracted from the $USERNAME special variable which is set when the user is authenticated.

7.2.8.7 Cached Query Changes

Cached Query Interaction With Read-Only Databases

In Caché 5.1, SQL queries that require Cached Queries will not work against read-only databases unless the ^mcq global is mapped to a database mounted as read-write. An example of this interaction is attempting to create a table in a read-only database using SQL DDL.

Cached Query Changes Are Not Journaled

In version 5.1, when cached queries are modified, the changes are no longer journaled. This prevents changes to cached queries inside a transaction from being written to the journal. Thus, shadowing will not apply cached query changes across systems.

Purging Cached Queries Is Immediate

Caché 5.1 no longer supports the concept of purging cached queries after *N* days, where *N* is a number of days defined in the configuration setting. When an application calls **Purge()**, it will purge all cached queries.

Cached Queries On Read-Only Databases

This version of Caché permits applications to Prepare and Execute Dynamic SQL cached queries that do SELECTs against the table of the database.

Note: Any attempt by the application to purge such queries will result in a <PROTECT> error. Purging cached queries requires write access to the database they are associated with.

Cached Query Purge Does Not Propagate To Other Systems

If a package is mapped to multiple namespaces within a single system, a compile/delete of a class in that package that has cached queries created by it will purge all the cached queries that use that class in each of the namespaces However, if the package mappings are to different machines via ECP and the class is recompiled/deleted, the purge of the cached queries will only occur on the system where the class is compiled. Cached queries form this class on other networked systems must be manually purged.

^mcq(“init code”) Execution Order Changed

The global node, ^mcq(“init code”), can be set to contain Caché commands to be executed when a connection is made via JDBC or ODBC. In Caché 5.0.12, the order of events during the connection was:

1. Read the connection info from the client
2. Run the code in ^mcq(“init code”)
3. Run the login code

In Caché 5.0.13, the sequence was changed to:

1. Run the code in ^mcq(“init code”)
2. Run the login code, including reading the connection info from the client

In Caché 5.1, this sequence is now

1. Run the login code, including reading the connection info from the client
2. Run the code in ^mcq(“init code”)

7.2.8.8 Ambiguous Names In SQL Queries

In FROM Clause

In previous versions of Caché, ambiguous field names in SQL queries were assumed to be associated with the earliest table mentioned in the FROM clause. This same situation in version 5.1 reports an error. The ambiguous names must be qualified to show their origin.

In ORDER BY Clause

Caché 5.1 reports an error if the field names in an ORDER BY clause are ambiguous, for example:

```
SELECT TOP 20
       %ID as ID,
       ((ID * 2) # 20) as ID
FROM Sample.Company
ORDER BY ID
```

This corrects a previously undetected bug in the query processor. Previous versions of Caché would associated the ambiguous name (ID in this case) with the last occurrence of that column name.

7.2.8.9 Privileges Required To Set Certain Options

You must have **%Admin_Security:Use** permission to execute the following SQL SET OPTION statements:

```
SET OPTION SUPPORT_DELIMITED_IDENTIFIERS = {TRUE | FALSE}
```

```
SET OPTION PKEY_IS_IDKEY = {TRUE | FALSE}
```

If you do not, the attempt to execute the statement will return an SQLCODE value of —99; this is a Privilege Violation error value. The reason is that these modify Caché configuration settings and you must be privileged to change them.

7.2.8.10 Changes To SQL GRANT And REVOKE

The SQL GRANT and REVOKE commands no longer support the following general administrative privileges:

- **%GRANT_ANY_PRIVILEGE**
- **%CREATE_USER**
- **%ALTER_USER**
- **%DROP_USER**
- **%CREATE_ROLE**
- **%GRANT_ANY_ROLE**
- **%DROP_ANY_ROLE**

In previous versions, SQL permissions were separately maintained. In version 5.1, these privileges are managed by Caché. SQL code which attempts one of these operations will be interpreted as granting a role having this name.

CREATE ROLE Details

In order to create a role definition through the SQL **CREATE ROLE** statement, the user must hold the **%Admin_Security:Use** privilege. If this privilege is not held by the user, an error -99 will be returned with an appropriate message.

DROP ROLE Details

In order to drop a role definition through **DROP ROLE**, at least one of the following must be true:

- The user has the **%Admin_Security:Use** privilege.
- The user is the owner of the role.
- The user was granted the role with the admin option.

7.2.8.11 SQL %THRESHOLD Removed

The **SQL %THRESHOLD** feature is no longer used in this version of Caché. Code that attempts to grant a threshold, for example,

```
GRANT %THRESHOLD ### TO SomeUser
```

will now receive an error at compile time. And code such as

```
REVOKE %THRESHOLD FROM SomeUser
```

will no longer revoke the threshold. The interpretation has changed; Caché 5.1 will attempt to revoke a role called **%THRESHOLD** from the user.

7.2.8.12 SQL Privileges On SAMPLES Granted To User _PUBLIC

At installation time, all SQL privileges for all tables, views, and procedures in the SAMPLES namespace are granted to the user named, _PUBLIC.

7.2.8.13 SQL Catalog Info For System Tables

The **SQLTables()** query of the %Library.SQLCatalog class returns a list of tables and views defined in the current namespace. In earlier versions, this list included System tables. In version 5.1, the System table information will only be included if the query is executed while in the %SYS namespace.

7.2.8.14 Collated Fields May Return Results In Different Order

Due to optimizations made in Caché SQL for 5.1, the results returned by queries on collated fields may be different. For example, consider

```
SELECT Dept, AVG(Salary)
FROM Personnel
GROUP BY Dept
```

where Dept is collated according to %SQLUPPER where the values entered in various rows are indiscriminate about case — some are uppercase, some lowercase, some with capital letters beginning each word, and so on. However, because of the GROUP BY clause, all departments are to be collected according to their value when converted to uppercase.

In prior versions of Caché, when this query's results were returned, the value of Dept returned was the actual value of one of the selected rows. In Cache 5.1, the value returned for Dept will always be represented in its collated form, in this case, %SQLUPPER.

This means that two queries such as

```
SELECT IdNum, Dept
FROM Personnel
```

and

```
SELECT Dept, COUNT(IdNum)
FROM Personnel
GROUP BY Dept
```

may not return the expected results. The first will return the actual values stored in the Dept column and the second will return those values converted to uppercase. This may not be what is desired by the application.

The prior behavior can be restored via the *%exact* qualification for Dept as in

```
SELECT %exact Dept, COUNT(IdNum)
FROM Personnel
GROUP BY Dept
```

7.2.8.15 Control Of Time Precision

There is a new SQL configuration setting which allows the specification of the precision of the time value returned by the **GETDATE()**, **CURRENT_TIME()**, and **CURRENT_TIMESTAMP()** SQL scalar functions. The default time precision can be set using the new API call:

```
PreviousValue = $SYSTEM.SQL.SetDefaultTimePrecision(value)
```

where *value* is the precision (the number of decimal places for the millisecond portion of the time value).

The default is 0; milliseconds are not returned in the values returned by these functions. The function returns the previous (or default) time precision setting. For example: After executing

```
Do $SYSTEM.SQL.SetDefaultTimePrecision(3)
```

GETDATE() will return a value in the format: 'YYYY-MM-DD HH:MM:SS.sss'. An application can still override this default by passing a specific time precision value to **GETDATE()**. For example: **GETDATE(5)** returns: 'YYYY-MM-DD HH:MM:SS.sssss'.

Note: This setting is used during the code-generation phase of the SQL engine. If you change the default time precision setting, you must purge any cached queries and recompile any class queries, embedded SQL routines, etc. for the new setting to take affect for that SQL statement.

CAUTION: While **CURRENT_TIME()** will return the time with a precision as specified in the default time precision setting, the LogicalToOdbc conversion of this time value does not support milliseconds. So if you have a default precision defined and **CURRENT_TIME()** is returned in a query through ODBC or JDBC, the milliseconds will be dropped from the value.

7.2.8.16 Owner Checked On DDL Create And Drop

When a users executes DDL to create or drop a procedure, query, or method in an existing class, Caché will not allow the action if the class has an OWNER defined, and the user is not the OWNER of the class.

7.2.8.17 ODBC & JDBC Permission Checking

Caché now checks the EXECUTE privilege for Stored Procedures invoked through ODBC and JDBC. A user may not call the procedure through ODBC or JDBC if the user has not been granted EXECUTE privilege on the procedure.

When looking at the list of procedures in the System Management Portal or from an ODBC or JDBC catalog query, the user will only see procedures that the user has privilege to call.

When creating a procedure through DDL, the creator user name is set as the default owner of the procedure. (This may be changed later by editing the class definition.) When the procedure is compiled, the owner of the procedure is granted EXECUTE privilege WITH GRANT OPTION if the owner does not have the %All role. If there is no owner specified in the class definition that projects the procedure, the owner is considered to be the user compiling the class.

When a procedure is dropped, or the class that contains the procedure definition is deleted, any execute privileges that had been granted on the procedure are dropped.

7.2.8.18 ^mdd Information Moved to ^oddEXTR

The *^mdd* global has been removed. In prior versions, this held SQL–related information. The information has been incorporated into the *^oddEXTR* structures.

You can mount an earlier database in Caché 5.1, but to use it you must upgrade it with the commands

```
Do $SYSTEM.OBJ.UpgradeAll()
Do $SYSTEM.OBJ.CompileAll()
```

After you run UpgradeAll and CompileAll, you cannot use the database in anything earlier than Cache 5.1.

7.2.8.19 Comparisons Involving NULL

This release of Caché corrects previous improper behavior in some SQL predicates involving constants and host variables whose values were NULL.

Application relying on the previous incorrect behavior of NULL testing for constants and host variables might have to be modified. This affects predicates of the form

```
field <> parameter
field > parameter
field >= parameter
```

where the value of "parameter" may be set to the NULL value. (Predicates involving the comparison operators "<" , "<=" , and "=" behaved correctly.) This means that existing predicates such as

```
field <> :hostVar
```

will have different behavior if :hostVar is bound to "" in COS. According to SQL three-state logic, this predicate should evaluate to NULL and fail rather than treating NULL as a value and succeeding for every field value other than NULL.

The previous behavior of a specific query could be restored, if necessary, by adding specific tests for NULL. AN existing query such as:

```
 field<>:hostVar
```

needs to be rewritten as

```
(field <> :hostVar OR (:hostVar IS NULL AND field IS NOT NULL))
```

to produce the same results as before.

7.2.9 System Error Code Changes

7.2.9.1 New Error Codes

This version of Caché adds news system error codes:

New System Error Messages

Error Code	Description
<ALARM>	An internal timer for user events has expired.
<COLLATECHANGE>	There was an attempt to change the collation algorithm while subscripted local variables are defined.
<DDP JOB OVERFLOW>	A Cache job's internal job number is greater than 1544 and it is attempting to access a DSM database using DDP. The job number is too large for DDP to handle.
<DSKFUL>	An attempt to write data to a disk file failed because the file reached its maximum size; some of the data was written but not all.
<EXTERNAL INTERRUPT>	Another process has attempted to interrupt this process.
<LICENSE ALLOCATION EXCEEDED>	This configuration has exceeded the number of license units it has been allocated from the pool of total units available.
<NETWORK UNLICENSED>	The application has attempted to access a remote directory, but there is no license for Caché networking.
<RESJOB>	An attempt was made to terminate a reserved job.
<TRANSACTION LEVEL>	The application has too many nested transactions pending.

7.2.9.2 Error Codes Removed

This release of Caché no longer supports the system error, <DISCONNECT>.

7.2.9.3 Globals Reorganized

Caché version 5.1 has reordered the subscripts in the globals that store user and system messages: ^CacheMsg and ^%qCacheMsg. The new order is domain, language, and id which allows subscript mapping of the message globals by domain.

The class dictionary version number is upgraded to 20 which will result in the user being asked to run **$SYSTEM.OBJ.Upgrade()** which will reorder the subscripts of existing ^CacheMsg globals. All error macros, routines and methods will keep the original arguments in the same order. therefore, no change in user code will be needed unless an application directly addresses the message global.

7.2.10 ObjectScript Changes

7.2.10.1 New System Variables

$USERNAME

In version 5.1, $USERNAME contains the name by which a user is known “inside” Caché for security purposes.

For example, suppose a user can successfully login to a Windows XP system with the username, “Smith”. If that user then attempts to access the Caché online documentation via the Cube, he or she is assigned the name, “UnknownUser”, for security purposes. If UnknownUser has no access to the online documentation, the user may be asked (depending on how Caché security is configured) to authenticate himself by supplying a userid and password known to Caché.

Caché 5.1 also retains the routine, **^%USER** that prints the name of the user running the current process as it is known to the operating system.

Note: ^%USER and $USERNAME are not required to be identical. The former results from the operating system login. The latter is based on a user successfully authentication to Caché security.

For example, suppose a user logs on Windows XP as user "Smith". However, if that user selects Documentation from the Caché Cube, the DocBook CSP application starts with a $USERNAME of "UnknownUser".

$ROLES

This variable contains a comma-separated list of all the roles held by the current user at any point during execution.

7.2.10.2 ObjectScript Compiler Upgrades

The ObjectScript compiler has been improved in version 5.1. As a result, it now generates code that cannot be run on previous releases. An attempt to do so will result in a <RECOMPILE> error.

The converse is not true. Compiled code from Cache 5.0 systems will run unchanged on version 5.1.

The following table gives the relationship between a version of Caché and a version of the ObjectScript compiler. The version number is made up of a “major” number and a “minor” number separated by a decimal point. The major and minor version of the ObjectScript compiler are returned by the ObjectScript functions **$ZUTIL(40,0,68)** and **$ZUTIL(40,0,69)**, respectively.

Caché Version	Compiler Version
3.2	9.6
4.0	9.7
4.1	9.7
5.0	10.0
5.1	10.1

A routine compiled on a version of Caché can be run on another version of Caché without re-compilation if

1. the major version of the compiler for each Caché release is the same, and
2. the compiler version of the system on which the routine will be run is greater than or equal to the compiler version of the system where the routine was compiled.

Note: The Caché Basic compiler uses the same version number as the ObjectScript compiler and is subject to the same compatibility rules.

CAUTION: This change means that ECP configurations are limited to having their servers on version 5.1 with clients on either version 5.0 or 5.1. ECP servers running Caché 5.0 cannot serve code compiled under version 5.1.

7.2.10.3 Permission Requirements For Some ObjectScript Commands

Because of their effect, some commands under certain circumstances now require the user to have specific permissions for them to succeed.

Command	Permission
Set	When applied to the special variable *$ROLES*, this is a privileged operation. Unless the application has the privilege to change *$ROLES*, the value will not be altered.
View	**%DB_<XXX>:R** is required to read blocks from the database; **%DB_<XXX>:W** is required to modify the database.

7.2.10.4 $ZTRAP Change

The reference material for $ZTRAP states that when the name of the error trap starts with an asterisk (*), it indicates that Caché should invoke the error handler at the context level where the error occurred. However, if the error trap is within the context of a procedure, then Caché cannot simultaneously establish the error context and the proper context for the local variables of the procedure. In Caché 5.1, the compiler has been changed to detect this usage and report it as an error at compile time.

Applications that wish to use this feature must ensure that either

- the subroutine containing the error recovery code is not a procedure, or
- the error handling logic is altered so it does not need to be run in the context of the error.

7.2.10.5 $ZERROR

Additional Information For Some Errors

In the event an error occurs, information about it is stored in the system variable, *$ZERROR*. In Caché 5.1, the string stored in *$ZERROR* includes more information than in previous versions.

For example, when a routine attempts to use a variable that has not been defined, *$ZERROR* now includes the name of the undefined variable. Whereas in previous versions of Caché the value of *$ZERROR* might look like this:

```
<UNDEFINED>zMethodName^Pkg.Class.1
```

in version 5.1, it looks generically like this:

```
<ERRCODE>Tag^Routine+line *someinfo
```

A consequence of this change is that error handling routines that made assumptions about the format of the string in *$ZERROR* may now require redesign to work as before. For example, the following will no longer work in version 5.1:

```
Write "Error line: ", $PIECE($ZERROR, ">", 2)
```

and should be changed to be something like

```
Write "Error line: ", $PIECE($PIECE($ZERROR, ">", 2), " ", 1)
```

The following table gives a list of errors that include additional info and the format of that information. The new info is separated from the previous text by a space.

Error Code	Description
<UNDEFINED>	the name of the variable (including any subscripts used)
<SUBSCRIPT>	the subscript reference in error
<CLASS DOES NOT EXIST>	the referenced class name
<PROPERTY DOES NOT EXIST>	the name of the referenced property and the class name it is supposed to be in, separated by a comma
<METHOD DOES NOT EXIST>	the name of the method invoked and the class name assumed to contain it, separated by a comma
<PROTECT>	the name of the global referenced and the name of the directory containing it, separated by a comma
<NOROUTINE>	the name of the routine being invoked

The names of variables local to routines (or methods) as well as the names of class properties and methods are indicated with an asterisk preceding the name. Global variable names are prefixed with a caret as expected.

Examples:

```
<UNDEFINED> *x
<UNDEFINED> *abc(2)
<UNDEFINED> ^xyz(2,"abc")
<PROPERTY DOES NOT EXIST> *SomeProp,Package.Classname
<METHOD DOES NOT EXIST> *AMethod,SamePackage.DifferentClass
<PROTECT> ^%GlobalVar,c:\cx\mgr\notyours\
```

7.2.10.6 $ZUTIL

Permission Changes

The authority needed to execute certain **$ZUTIL** functions is more specific in version 5.1 than prior releases. Unless otherwise noted in the following table, published **$ZUTIL** options require no special permission.

Function Number	Requirement	Explanation
5	**%DB_<XXX>:R**	Subject: change to another namespace In version 5.1, changing to a different namespace requires that the user have **%DB_XXX:Read** where XXX is the name of the database that contains the globals for the namespace.
58	N/A	Subject: set the privilege level required to use the **XECUTE** command. This function has been removed in Caché 5.1.
69	**%Manager**	Subject: set system-wide defaults This function must be invoked from a non-edited routine in the CACHESYS database and not part of any indirection; or else be in a job that holds the WRITE permission on the CACHESYS database.
78	**%Manager**	Subject: search journal file for open transactions. This function must be invoked from a non-edited routine in the CACHESYS database and not part of any indirection; or else be in a job that holds the WRITE permission on the CACHESYS database.
130	**%Manager**	Subject: set or return the domain ID or index This function must be invoked from a non-edited routine in the CACHESYS database and not part of any indirection; or else be in a job that holds the WRITE permission on the CACHESYS database.
131	N/A	In previous releases, the subfunction, 1, returned the system identifier string consisting of the current system name followed by a colon (:), the IP address (Windows) or MAC address (UNIX) followed by a comma (,), and the pathname of the mgr directory. In version 5.1, it returns the name of the system, followed by a colon (:), and the name of the Caché instance that is running.

$ZUTIL(4)

This function is used to stop processes in Caché. In version 5.1, it has been refined to protect system jobs from interference by unprivileged applications. For example, **$ZUTIL(4, <pid>)** will no longer terminate a daemon. Instead, it will return an error status of 0.

Moreover, a process which is exiting and running %HALT, or any of its subroutines such as %ROLLBACK, will not respond to this function. The process issuing the RESJOB will now receive an error status of -4 meaning that the target ignored it.

Shutting down the system with "ccontrol stop" will terminate these processes as it has in the past. This can also be done in version 5.1 with the function, **$ZUTIL(4, <pid>, -65)**.

CAUTION: When , **$ZUTIL(4, <pid>, -65)** is used for this purpose, any open transactions will not be rolled back even though the locks which protected it will be released.

$ZUTIL(49)

The information it returns has been extended to better describe the database:

- $ZU(49, <*sfn*>, 3) — Version 5.1 adds several fields to the end of the previously returned information:
  - <*SysNumber*>: the remote system number, or zero if the database is local
  - <*DirPath*>: the database path on the system
  - <*ResourceName*>: the resource associated with the database
  - <*BlockSize*>: the database block size in KB
  - <*Collation*>: the database collation
  - <*DirectoryBlock*>: the DB directory block number. For ECP clients this is the local cache directory block number of the database in CacheTemp. The ECP server does not send the database directory block number to the clients.

  All the values returned are separated from each other by "^".

7.2.10.7 $ZF

Permission checking is also applied to some operations of $ZF; the following table lists the permission needed for those whose execution is restricted.

Function Number	Permission	Explanation
—1	**%System_Callout:U**	Executes a program or command as a spawned child process and waits for the child process to return.
—2	**%System_Callout:U**	Executes a program or command as a spawned child process and returns immediately.

The other **$ZF** functions remain unprivileged operations as in previous versions of Caché.

7.2.11 Storage Changes

7.2.11.1 %CacheStorage Changes

New Property Method: GetStored()

A new property method is now implemented for all storable properties of persistent classes that are using default storage (%CacheStorage). It is **<propertyname>GetStored()**. This method accepts an object ID value (not an OID) and returns the logical value of the property as stored on disk. If **<property>StorageToLogical()** is used then it is applied to convert the stored value to a logical value.

This method is not valid for collections stored in a subnode structure. If an object identified by the supplied ID does not exist, an error will be reported. This method is not implemented for properties that are not stored: transient, multidimensional, or calculated properties. In these cases, Caché will report a <METHOD DOES NOT EXIST> error.

ID Counter Check Available With Default Storage

A new API function to check some system assigned object ID counters is now available in this version. An id check expression is generated by the compiler for each class using default storage with system assigned id values. (Child classes using default storage and system assigned IDs do not have this expression.)

The function, **$$CheckIDCounters^%apiOBJ(.errorlog)**, will examine all extents in the current namespace. If an idcheckexpression is found, it will be invoked. The id check expression will fail if the last id in the extent has a value higher than the id counter location. In this case, an entry is placed in the errorlog array, subscripted by extent name. The id check expression is also included in errorlog so that the user can repair the problem.

An application should invoke this utility with:

```
Set sc = $$CheckIDCounters^%apiOBJ(.errarray)
```

After the utility returns, *sc* will be set to a standard status message. If errors were found, they will be stored in the multidimensional array, *errarray*.

%ExistsId() Is Now Generated For Default Storage

In Caché 5.1, this generated method will now validate the id value passed. If any components of the id are null, %ExistsId() will return zero (the object does not exist). If all components are not null, then the object reference will be checked for existence.

Note: This method is meant to be called in the class that is an extension of %Library.Persistent. Passing in an ID value that is not constructed by the class of which it is an instance not recommended as it breaks encapsulation.

7.2.11.2 %CacheSQLStorage Changes

Valid Row References Required For $PIECE Access Types

Applications using %CacheSQLStorage that employ two or more subscript levels of Access Type, **$Piece**, and have specified Data Access expressions for those subscript levels, must supply a valid Row Reference in the map definition. This version of Caché will no longer automatically generate one under these circumstances.

New Dynamic Value Substitution

Caché now supports the use of

- *{%%CLASSNAME}*: expands to the name of the class without quotes, for example,

  ```
  Do ##class({%%CLASSNAME}).MyMethod()
  ```

- *{%%CLASSNAMEQ}*: expands to the name of the class within quotes,

  ```
  Set ThisClass = {%%CLASSNAMEQ}
  ```

- *{%%TABLENAME}*: expands to the name of the table within quotes,

  ```
  Set MyTable = {%%TABLENAME}
  ```

in the following locations within a %CacheSQLStorage map definition:

- Map Subscript
  - Data Access expression
  - Invalid Conditions
  - Next Code
  - Access Variable Expressions

  Map Data

- Retrieval Code

7.2.12 Java

7.2.12.1 Package Names May Not Be SQL Reserved Words

If a Caché class is to be projected to Java, and any component of the package part of the projected class name matches an SQL reserved word (ignoring case), the attempt to project the class will report an error that the metadata for the Java class is missing its column names. This error can be avoided by using package names that are not the same as any SQL reserved word.

7.2.13 Caché TERMINAL

7.2.13.1 Terminal Is Always Unicode Now

There is now only one version of TERMINAL which runs internally using Unicode characters. By default, it starts with the ISO network encodings "Local Encoding 2" and "Network Encoding 2". In order to display characters > 255 you must change the encoding to UTF8. As a result of this enhancement, the "Pass 8–bit Characters" setting has been removed.

When there are multiple instances of Caché, some Unicode and some 8 bit, it is good practice to set the encoding explicitly for each TERMINAL instance. Then the defaults no longer apply.

7.2.13.2 Argument Changes

Caché TERMINAL no longer supports command arguments /size, /pos, and /ppos. It has been enhanced to handle characters internally in Unicode and provide for the proper translations to and from servers in different locales.

7.2.13.3 Password Echo

In previous releases, when the TERMINAL prompted for a password, it did not echo any characters to the output device. As of version 5.1, when Caché password login is enabled, each character of the password will be echoed as an asterisk (*). Any application that performs a login supplying a userid and password at the TERMINAL prompt must be made aware of the echoing behavior if it does pattern matching on the characters TERMINAL transmits.

7.2.13.4 Launching From The Windows Cube

In this version of Caché, the way the Cube determines whether to use Telnet or TRM for a remote TERMINAL session has changed. Servers are placed in the list displayed under the "Remote System Access" menu according to these rules:

1. Remote servers are always shown as enabled.
2. Local servers where the IP address of the server is not that of the local host, and the server name is not a local instance name will be treated like remote servers because the server is not associated with a local instance.
3. Local servers (where the IP address is the local host address, and the server name is the same as a local instance name) will be grayed if the configuration is down or telnet is disabled for that instance. Otherwise the server name will be enabled.
4. A telnet connection will always be available when using the Remote System Access menu to launch a terminal.

When the terminal is launched from the main cube menu:

- If the active preferred server is associated with that instance, a private terminal connection will be made. The ttle bar for the TERMINAL windows will contain “TRM” followed by the process id and the name of the Caché instance. If an instance is not running, the instance will be started before launching the terminal.
- Otherwise, a telnet connection will be made. The terminal's title bar will contain the hostname followed by “NTI - Cache Telnet". The cube never starts an instance of Caché different from the one it was installed with.

7.2.13.5 Local And Network Encodings Are Now Distinct

The local and network translation settings for Caché Terminal are now stored separately for 8-bit and Unicode installations to permit the user to choose different behavior for Unicode and 8-bit installations which may exist on the same host. In prior versions, they had been the same.

7.2.14 SOAP Parameter Location Changes

This version changes the location of the parameters that control SOAP logging behavior. In previous versions these were in **^%SYS**. In 5.1, they reside in the namespace from which the SOAP request is made. The parameters at issue are:

- ^ISCSOAP(“Log”) — set to “1” when web services requests and client responses should be logged
- ^ISCSOAP(“LogFile”) — the full pathname of the file where the logged information is placed

7.2.15 Callin And Callout

7.2.15.1 On The Windows Platform

On Microsoft Windows, Caché is now compiled with Visual Studio 7.1. User applications communicating with Caché using the CALLIN or CALLOUT interfaces must be upgraded to this version of Visual Studio.

7.2.16 CSP Changes

7.2.16.1 CSP Grace Period Changed

As part of the licensing changes introduced with Caché version 5.1, how CSP treats sessions has changed.

If a CSP session visits more than one page and the session is ended either from a session timeout or from the application setting %session.EndSession=1, CSP will release the license immediately rather than adding on an extra grace period.

If the session is just active for a single page, CSP will hold the session open for a five–minute grace period when the session is ended.

7.2.16.2 CSP Page Timing Statistics Default To Off

In Caché 5.1, the class parameter, *PAGETIMING*, has been changed to have a default value of zero. In earlier versions, its default value was 1. The zero value turns off the collection of page timing statistics for all classes that inherit from it. If an application relies on the page timing statistics being collected for CSP pages, then it will need to be modified to inherit from a superclass that has *PAGETIMING* set to 1.

7.2.17 Collation For Locales Now On By Default

Please refer to the discussion in the Administrator section.

7.2.18 Caché Dreamweaver Extension Revised

The Dreamweaver extension has been extensively revised to improve its security in version 5.1. It now uses the C++ binding exclusively. Users who wish to use this extension must have the **`%Development`** privilege. The extension will continue to work for those users without this privilege but no data from Caché will be visible in accordance with our security rules.

As a result of this change, the following must be true:

- The Dreamweaver extension now requires Cache 5.1 on the server for operation.

- The following directory must be present in the %path% environment variable,

  ```
  \Program Files\Common Files\Intersystems
  ```

7.3 Operators

7.3.1 System Management Portal

Prior to this version, how Caché was administered depended heavily on the platform where Caché ran. With version 5.1, InterSystems introduces a new administrative interface that is common across all platforms. Caché 5.1 now uses a browser-based interface, the System Management Portal, for system management.

A brief summary can be found in the Administrator section of this document and more complete information on the System Management Portal can be found in the System Administration documentation.

7.3.2 PERFMON And %SYS.MONLBL Coordination

These utilities each use some of the same Caché data structures for gathering data. So they should not execute at the same time; otherwise there is a risk that they may compromise each other's data. In version 5.1, program checks have been added to prevent their simultaneous execution.

7.3.3 Backup Information Changes

Beginning with version 5.1, the location where the backup database list is maintained has been changed. ^SYS("BACKUPCHUI") is no longer used. The list is maintained by the methods, **Backup.General.AddDatabaseToList()** and **Backup.General.RemoveDatabaseFromList()**. Moreover, InterSystems strongly recommends against setting it manually since this works at cross-purposes with the methods. Use the System Management Portal or the **^BACKUP** utility instead.

7.3.4 New Question In Journal Restore

In prior versions, the journal restore routine, **^JRNRESTO**, did not properly handle the restoration of journal files written on different operating systems. The error occurred in the handling of directory names specified by those systems.

Caché version 5.1 now accounts for this by asking whether the journal was produced on a different kind of operating system. However, if a site is using a script to drive the journal restore, the script will have to be modified to provide an answer to the new question.

7.3.5 Cluster Member Startup Improved

The logic for a Caché instance to join a member of a cluster has been improved to avoid confusion between systems making up the cluster. For details, please see the Administrator portion of this book.